On the Complexity of Verifiable Secret Sharing and Multiparty Computation

Ronald Cramer[*], Ivan Damgård, Stefan Dziembowski [†]
Aarhus University, BRICS[‡]
{cramer|ivan|stefand}@daimi.au.dk

(This paper appears in Proc. of STOC 2000)

Abstract

We first study the problem of doing Verifiable Secret Sharing (VSS) information theoretically secure for a general access structure. We do it in the model where private channels between players and a broadcast channel is given, and where an active, adaptive adversary can corrupt any set of players not in the access structure. In particular, we consider the complexity of protocols for this problem, as a function of the access structure and the number of players. For all access structures where VSS is possible at all, we show that, up to a polynomial time black-box reduction, the complexity of adaptively secure VSS is the same as that of ordinary secret sharing (SS), where security is only required against a passive, static adversary. Previously, such a connection was only known for linear secret sharing and VSS schemes.

We then show an impossibility result indicating that a similar equivalence does not hold for Multiparty Computation (MPC): we show that even if protocols are given black-box access for free to an idealized secret sharing scheme secure for the access structure in question, it is not possible to handle all relevant access structures efficiently, not even if the adversary is passive and static. In other words, general MPC can only be black-box reduced efficiently to secret sharing if extra properties of the secret sharing scheme used (such as linearity) are assumed.

[*] Work done while employed at ETH Zurich, Comp. Sc. Dept.
[†] On leave from Institute of Informatics, Warsaw University, Poland.
[‡] Basic Research in Computer Science, center of the Danish National Research Foundation

1 Introduction

In this paper, we consider three related problems, namely *secret sharing* (SS), *verifiable secret sharing* (VSS) and *multiparty computation* (MPC).

SS was introduced by Shamir[16] and generalized by Itoh et al.[11]: a *Dealer* has a secret s and distributes a set of *shares* $s_1, \dots, s_n$ to n players, such that s can be reconstructed only by certain *qualified* subsets of players while unqualified subsets have no information about s. The collection of qualified sets is called the *access structure*. We stress that we consider here secret sharing for general access structures, rather than threshold schemes where the access structure may only consist of all sets of size larger than some threshold. It is assumed that the dealer computes the shares correctly, and that players input correct shares for reconstruction.

When these assumptions are dropped, we get the (seemingly) harder problem of VSS (Chor *et al.* [4]): here, some of the players, including the dealer, may not follow the protocol. It may even be the case that some of them turn bad dynamically as the protocol proceeds, as long as the total set of bad players remains unqualified. Still the remaining honest players should be able to verify that they have shares of a well defined secret, while the cheating players should get no information about it, if the dealer is honest. Finally, the honest players should be able to reconstruct the secret, even against the actions of the cheating players.

Our final, still more general problem is that of MPC (A. Yao [17], Goldreich, Micali and Wigderson [9]): here, all players have a secret input, and the goal is to compute an agreed functions of these inputs, while maintaining privacy of the inputs and correctness of the result, again assuming that the set of bad players at any given time is unqualified.

The classical results in unconditionally secure VSS/MPC by Ben-Or, Goldwasser and Wigderson[2], Chaum, Crépeau and Damgård [3] and Rabin and Ben-Or [15] can be seen as results that build efficient VSS and MPC protocols based on Shamir's threshold secret sharing scheme, in the model where secure channels are assumed to exist between every pair of players.

Gennaro [8] was the first to consider VSS secure for general access structures. Then Hirt and Maurer [10] characterized exactly those general access structures for which VSS and MPC are possible.

Continuing the line of research from [2, 3, 15], Cramer, Damgård and Maurer [6] have shown that VSS and MPC for general access structures can be built efficiently on top of any linear SS scheme (see also [5]). Thus a natural final step is to ask what happens if we start from an *arbitrary* SS scheme?

Informally, what we show in this paper is that VSS is as easy to achieve efficiently as ordinary SS, more precisely, there exists an efficient reduction that builds a secure VSS protocol from any SS scheme secure for the same access structure, provided VSS is possible at all for that structure. Since VSS trivially implies SS, this is an optimal result.

Similarly, showing that MPC in this sense is no harder than SS would be an optimal result. However, we show an impossibility result indicating that there is not much hope of proving this. A reduction showing how to do secure MPC for some access structure given *any* SS-scheme for that structure cannot make any assumptions on the way the SS-scheme works. So the natural approach is to treat the secret sharing as a black box, relying only on the functionality that follows from the definition of secure SS. We show that if we restrict ourselves to such reductions, there are access structures that cannot be handled efficiently, where by "efficiently", we mean that protocols run in time polynomial in the number of players, counting usage of the SS-scheme as only one step.

Thus, general reductions building MPC from SS would have to either depend on the particular kind of SS-scheme being used (such as reductions depending on linearity of the SS scheme) or be inefficient on some access structures. This may be seen as an indication that, as far as applicability to unconditionally secure MPC is concerned, there is a fundamental difference between linear SS schemes and general ones. By contrast, it is shown in[6] that general SS does suffice for *computationally* secure MPC.

2 A More Detailed View

To explain our results in more detail, we need to describe more precisely the model we use: we have a set of n players, connected pairwise by private channels, moreover, a broadcast channel is also available. We are given a *monotone access structure* Γ, that is, Γ consists of subsets of the player set, such that $A \in \Gamma$ and $A \subset B$ implies $B \in \Gamma$. For instance, Γ could consist of all subsets of size $n/2$ or more. For convenience in the following, we will in stead talk about the family $\mathcal{A}$ of subsets not in Γ. Such a complement of an access structure is known as an adversary structure, a general notion introduced by Hirt and Maurer[10].

Finally we have an *adversary* who can corrupt any subset A of players, as long as $A \in \mathcal{A}$. This is called an $\mathcal{A}$-adversary. The adversary may be *passive*, meaning that he just gets access to all data of corrupted players, or *active*, meaning that he takes control over corrupted players and may make them deviate from the protocol they were supposed to follow. Orthogonal to this, we distinguish between *static* adversaries who must decide before the protocol who to corrupt, and *adaptive* adversaries [15] who may decide dynamically throughout the protocol whom to corrupt, as long as the total corrupted set is in $\mathcal{A}$.

In this model, security of secret sharing (SS) can be rephrased to the requirement that a passive, static adversary gets (almost) no information about the secret when it is distributed. And that the secret can be reconstructed, even if the adversary can make the corrupted set of players fail to input shares for reconstruction. Note that since VSS protocols in our model usually have non-zero error probability, we will allow SS schemes in our definition to also have non-zero error, in order to make a reasonable comparison of the two problems.

In order to talk about the complexity and error probability of an SS scheme, we will think of it as two probabilistic algorithms $Distr$ and $Recon$, where $Distr$ gets as input the secret s, the number of players n, and a security parameter k as input, it then computes a set of n shares as output. Sharing s means that one of the players, the dealer, runs $Distr$ and sends the shares privately to the players. For reconstruction, some subset of the shares are broadcast, and each player can run $Recon$, which gets as input the subset of the shares, n and k, and outputs a value s'.

Thus, underlying this, we have not just one, but a family of adversary structures, one for each n. We will say that $(Distr,\ Recon)$ is secure against the family $\mathcal{F} = \{\mathcal{A}_n\}_{n=1..\infty}$ if the following hold for any n and any static, passive $\mathcal{A}_n$-adversary:

- Assuming the dealer is not corrupted, the amount of Shannon information that the adversary gets about the secret after it is shared is negligible in k.
- Sharing s and running $Recon$ on input a set of shares not in $\mathcal{A}_n$ results in output s, except with probability negligible in k. Running $Recon$ on input a set of shares in $\mathcal{A}_n$ results in output a special symbol $\perp$, indicating that the input set was unqualified, again except with probability negligible in k.

Here, negligible in k means that the quantity converges to 0 as a function of k faster than any polynomial fraction. Note that it would not make any essential difference if the behavior of $Recon$ on unqualified sets was left undefined: one can always test, with arbitrarily small error, if a set is qualified by sharing a random secrets and testing if they can be reconstructed from the subset in question.

Going to VSS, we will think of this as first a protocol for distribution that takes a security parameter k as input, and where one of the players, the dealer, gets the secret as private input. Secondly, there is a protocol for Reconstruction, where each player starts from his view of the distribution (which we think of as his share), and reconstructs a value for the secret. Such a VSS is secure against the family $\mathcal{F} = \{\mathcal{A}_n\}_{n=1..\infty}$ if the following hold for all n and all adaptive, active $\mathcal{A}_n$-adversaries:

- After distribution, a secret is uniquely defined from the views of the set of incorrupted players, except with probability negligible in k.
- If the Dealer remains uncorrupted, the adversary has a negligible (in k) amount of information about the secret.
- Reconstruction of the secret results in all uncorrupted players reconstructing the secret defined at distribution time, except with negligible probability in k.

Hirt and Maurer [10] show that these criteria can be met if and only if the adversary structure is $Q2$: for any two sets in the structure, their union is not the whole player set. The obvious question is therefore: when can it be done *efficiently*? – which we here will take to mean in polynomial time in the number of players. The number of $Q2$ structures is doubly exponential in n (see [10]) so it follows from a counting argument that we cannot hope to handle all structures efficiently.

One way we could hope to get upper and lower bounds for VSS is by relating it to the simpler problem of SS. Cramer, Damgård and Maurer [6] show that any *linear* secret sharing scheme implies a VSS with polynomially related efficiency [1]. Here, a linear secret sharing scheme is one in which the shares and the secret are elements in a finite field, and the secret can be obtained as a linear function of the shares. Such schemes can be based on monotone span programs [13, 1].

Our first main contribution is a similar result, that holds for *any* secret sharing scheme. Since VSS trivially implies SS without significant loss of efficiency, this is the best result we can hope for.

THEOREM 1 *Given any secure secret sharing scheme $\mathcal{S} = (Distr, Recon)$, secure against a family $\mathcal{F}$ of $Q2$ adversary structures, there exists a VSS protocol secure against $\mathcal{F}$ with complexity polynomial in n, k and the running time of $\mathcal{S}$.*

We show this by combining the information checking idea of [15, 5] with a new technique for upgrading from static to adaptive security.

Note that one way to prove such a result would be to provide a kind of "efficient compiler" that takes as input the algorithms $(Distr,\ Recon)$ and produces a VSS protocol for the same adversary structure, with polynomially

[1] In fact, they worked in a model with no broadcast and zero error but a stronger condition on the adversary structure. However, the result translates to our model using the techniques of [5]

related efficiency. We prove a slightly stronger result, in that we construct a single VSS protocol that works when given only black-box access to the algorithms of any SS scheme.

A seemingly even harder problem than VSS is that of MPC, as described above. For this problem a set of results very similar to those for VSS is known: In was proved in [10] that the $Q2$ condition on the adversary structure is necessary and sufficient for MPC to be possible. In [6] it is shown how to perform secure distributed multiplication efficiently when given a linear SS scheme, and how efficient MPC follows from this given also a commitment scheme with extra homomorphic properties. Using a VSS construction from [5] as commitment and exploiting its linearity, it then follows that a linear SS scheme for a $Q2$ adversary structure implies an MPC protocol for the same structure with polynomially related efficiency, secure against an active, adaptive adversary.

So it is natural to ask whether a result similar to Theorem 1 holds for MPC? As for VSS, this is the best result one can hope for in terms of poly-time efficiency. However, we show a result implying that a reduction of the type we provided to prove Theorem 1 does not exist for MPC, not even if we assume the adversary is passive and static. Informally, we show that MPC protocols which get black-box access for free to secret sharing, but do not use any special properties of the SS scheme in question, cannot handle all $Q2$ adversary structures efficiently.

Since, in both the passive and active models, distributed addition is easily handled (as we will argue later on), it follows that it is essentially distributed multiplication, or equivalently, Oblivious Transfer, that prohibits efficient construction of MPC protocols from black-box SS-schemes.

To make this more precise, we first fix (for concreteness) a function which will turn out to be hard to compute securely for all $Q2$ structures, namely the function f_{AND} which is the AND of n input bits, one from each player.

We will allow protocols to be constructed non-uniformly over n, k, this only makes the impossibility result stronger. Thus for a fixed value of n, k, we can write down the computing done by each player at each round of the protocol as a Boolean circuit. Sending of messages translates directly to wires connecting these circuits, and the view of a player becomes the collection of values that are handled by his part of the circuit. In the following, we will only be interested in what happens when both n and k go to infinity, so for simplicity, we only look at cases where $n = k$.

So we define a protocol for computing f_{AND} for an arbitrary number of players n as a family of Boolean circuits of this type $\{C_n | \ n = 1, 2, ..\}$, where C_n specifies the actions of the protocol for n players (and security parameter $k = n$). A protocol is polynomial time, if each C_n has size bounded by some polynomial.

We will say that a protocol computes f_{AND} securely against a family $\mathcal{F} = \{\mathcal{A}_n\}_{n=1..\infty}$ of adversary structures, if the following two conditions hold for all static, passive $\mathcal{A}_n$-adversaries [2]:

correctness For any set of inputs bits $b_1, ..., b_n$, the protocol computes as result for all players $b_1 \wedge ... \wedge b_n$, except with negligible probability (in $n = k$).

privacy For $A \in \mathcal{A}_n$, if the adversary corrupts A, he learns nothing about inputs bits outside A, beyond what is implied by input bits in A and $b_1 \wedge ... \wedge b_n$ (except for an amount negligible in $n = k$).

Note that requiring only passive, static security makes the impossibility result stronger.

Finally, we discuss how to model protocols that are allowed to use SS secure against the adversary structures in some family $\mathcal{F} = \{\mathcal{A}_n\}_{n=1..\infty}$ but without relying on which particular SS-scheme is used.

Note that it would not work to give the protocol black-box access to the algorithms $(Distr, Recon)$ of some scheme. The shares thus produced depend of course on the algorithm, so as a result the protocol might still take actions depending on the particular scheme used. To avoid this, we give in stead the protocols access to an idealized secret sharing where the shares are replaced by a random number that is unrelated to any particular SS scheme.

More precisely, we allow protocols to make use of an extra uncorruptible player T with unbounded computing power called an SS-$\mathcal{F}$-*oracle*. T will implement an "ideal" SS w.r.t. $\mathcal{A}_n$ whenever there are n players: any player can send a message "share s" to T containing a secret s. Then T will remember s and distribute to all play-

[2] Usually, when defining security of multiparty computation, one cannot separate correctness and privacy as we do here. However, in our particularly simple case of static and passive security, this is not a problem

ers a randomly chosen but unique number $ID(s)$. At any later time, the protocol can issue a request to T "Reconstruct A, $ID(s)$" meaning that the players in the subset A would like to reconstruct the secret identified by $ID(s)$. Now, T computes whether A is in $\mathcal{A}_n$. If not, T will send s to the players in A. Otherwise T will only say that A is in $\mathcal{A}_n$.

When measuring running time, access to T counts only as one step - the protocol is not charged for the internal computing done by T [3]. A protocol with such an extra player is called an *SS-oracle protocol*. Such a protocol is said to compute a function securely against $\mathcal{F}$ if it can do so when given access to an SS-$\mathcal{F}$-oracle. Our result now is:

THEOREM 2 *There exist families $\mathcal{F}$ of $Q2$ adversary structures, such that no polynomial-time SS-oracle protocol computes f_{AND} securely against $\mathcal{F}$.*

Note that if we were talking about protocols with no oracle access, we would have a lemma already shown in [10] for which a simple counting argument suffices: Let a maximal $Q2$ adversary structure be one to which we cannot add a new subset without loosing the $Q2$ property. It is then easy to see that there are doubly exponentially many maximal $Q2$-adversary structures on n players. On the other hand, there are only exponentially many different protocols that can be specified by a number of bits polynomial in n. Thus, if the result was false, there would exist some protocol that could handle several different access structures. But the same protocol cannot be secure for two different maximal $Q2$ structures because it would then be secure for their union, which is not $Q2$.

However, in our case this argument breaks down: our protocols have access to an oracle giving answers that depend on the access structure in question, thus an oracle protocol may take different actions for different access structures. The main technical problem we solve is to show that even this is not sufficient to do MPC efficiently for all structures.

Note that Theorem 2 does not rule out that a result similar to Theorem 1 could hold for MPC: it may be the case that for every (class of) SS-scheme(s) there exist MPC protocols with polynomially related efficiency that depend on the particular scheme, or class of schemes considered. Only the existence of a general black-box reduction is ruled out.

[3] thus, in our circuit model, a call to T is modeled as a single oracle gate doing internally all T's computation.

3 Distributed commitments

Informally, a commitment scheme consists of a commit protocol, where one of the players, the committer, on input some value a to commit to, distributes some information to the other players, possibly also some protocol is carried out to verify this information. The total information distributed is called the *commitment* and is denoted $[a]$ (suppressing random coins used in computing the commitment). Assuming the committer is not corrupted, the adversary should get only negligible information on a. If the committer is corrupted, he may succeed in distributing a commitment that is not well formed, we will denote such a *bad* commitment as $[\cdot]$.

Secondly, there is a protocol for opening a commitment, where all honest players either compute the same value or they all reject. If the committer stays incorrupted, then all incorrupted players compute a as the result of opening $[a]$. If the committer is corrupted $[\cdot]$ will always be rejected, and opening $[a]$ is either rejected or a is computed.

As usual, these conditions should hold, except with negligible probability.

3.1 Static Adversaries

For commitments secure against a static adversary, we can generalize easily a commitment scheme from [15] (called weak secret sharing (WSS) there).

A basic tool introduced in [15], with efficiency improvements in [5], is Information Checking. This tool allows a sender S to give a message m to a receiver R (and send related information to other players) such that first, the adversary will not learn m if S, R are incorrupted, and secondly, even if S is corrupt, an honest R can later broadcast m and convince all other honest players that the broadcasted value was really the one received from S. This also means, of course, that a corrupt R cannot convince the players about a message m', if he actually received $m \neq m'$ from an honest S.

These properties hold unconditionally, with a negligible error probability. Clearly, these are the essential properties that ordinary digital signatures could ensure, based, however, on computational assumptions.

For simplicity, we will describe the following protocol using the terminology of digital signatures. Substituting Information Checking will then give unconditionally secure protocols for the same purposes.

Given an SS scheme $(Distr, Recon)$, a commitment scheme can be built as follows:

- To commit to a, the committer shares a using $Distr$, to get shares $a_1, \ldots, a_n$. He signs each share and distributes the shares and signatures to the players.
- To open a commitment, the committer broadcasts a, all the shares, and the random input used when creating the shares. Each player P_i checks by running $Distr$ that the broadcasted shares are consistent with a and that a_i matches the shares he received originally. If not, he complains and broadcasts the original share and signature. If a properly signed share is broadcast that does not match the committers broadcast, the opening is rejected. Otherwise it is accepted.

Note that the use of the IC-scheme gives rise to two relevant error probabilities in this scheme. First, the probability e_0 that the adversary can successfully produce a forged signature in the opening phase (assuming that the committer is honest), and second, the probability e_1 that a signature broadcast by an honest player in the opening phase is not accepted (assuming that the committer is corrupted). The parameters of the IC-scheme can easily be set such that both error probabilities are negligible as a function of a security parameter k and such that the complexity of the scheme is polynomial in n and k.

We now analyse the statically secure commitment scheme. Clearly, since the adversary can only corrupt an unqualified set, he does not learn a, if the committer is honest.

And since the adversary structure is $Q2$, the complement of the set of corrupted players is qualified, so one of the following two cases occur: either the commitment is good, i.e. all honest players hold consistent shares of some value a. Then even a corrupt committer cannot open a different value without changing the share of at least one honest player, which leads to reject in the opening, except with probability e_1. Or the commitment is bad, which means opening it will be rejected, except with probability e_1.

However, this argument only works because the set of corrupted players is fixed. Consider what happens if the adversary is adaptive. Clearly, a good commitment stays good. But a bad commitment may become good as more players get corrupted. However, depending on which players are corrupted, a commitment $[\cdot]_i$ may turn into a good commitment in several different ways, defining different values. This problem was observed for the case of threshold secret sharing in [5], and it was shown there how an adaptive adversary can break the commitment scheme.

The attack generalizes easily to many non-threshold structures, namely those that are $Q2$, but not $Q3$ [4]. Let A, A', B be a disjoint partition of the player set $\mathcal{P}$. It is not difficult to see that it is possible to construct $\mathrm{s} = (s_1, \ldots, s_n)$ such that when restricted to the players in $A \cup B$, this is a set of shares uniquely consistent with a value a, whereas restricted to $A' \cup B$, it is uniquely consistent with a different value a', and such that both sets of shares have the right distribution: compute shares in a, and delete the shares intended for A'. Since B is non-qualified, its shares can be extended (with the right distribution) to a full set consistent with a'. Finally, delete the shares for A from this extension. Note that since the adversary structure is Q^2, both $A \cup B$ and $A' \cup B$ are qualified sets.

If in the commit phase the committer uses s constructed in this way instead, he can later open as a if he corrupts the players in A', and as a' if he corrupts the ones in A.

So to obtain adaptive security, we need to solve this problem. A solution was given in [5], which however requires that the underlying SS scheme be linear.

3.2 Upgrading to Adaptive Security

The basic idea to get adaptive security is to build on top of our statically secure commitment scheme from Section 3.1 a new one with the added property that we can ensure at commitment time that a commitment is good.

[4] A $Q3$ adversary structure is one for which no three sets in the structure cover the entire set of players

For simplicity, we will describe a bit commitment scheme in which only a single player V verifies that the committer C has created a good commitment. We then later describe how to use this to build protocols convincing all honest players.

We assume without loss of generality that our statically secure commitment scheme allows a player to commit to any $2n+k+1$ bits string, [5] and we set the error probabilities [6] e_0, e_1 in the statically secure commitment scheme such that both are at most 2^{-k-1}.

A commitment to bit b in the new scheme is denoted $[[b]]_{C,V}$.

Protocol Commit

1. To make a commitment $[[b]]_{C,V}$, the committer C chooses a random $2n+k+1$ bits string $\mathbf{a}$, and commits to it using the statically secure commitment scheme, resulting in $[\mathbf{a}]$.
2. The verifier V randomly chooses two distinct $2n+k+1$ bit strings $\mathbf{v}_0, \mathbf{v}_1$, and broadcasts them.
3. C computes the string $\mathbf{z} = \mathbf{v}_b \oplus \mathbf{a}$, and broadcasts it. We set $[[b]]_{C,V} = ([\mathbf{a}], \mathbf{v}_0, \mathbf{v}_1, \mathbf{z})$.

Protocol Open

1. C broadcasts the bit b, and opens the commitment $[\mathbf{a}]$.
2. Assuming that the opening in the previous step was accepted, V accepts the opening of $[[b]]_{C,V}$ if it indeed holds that $\mathbf{z} = \mathbf{v}_b \oplus \mathbf{a}$. Otherwise it is rejected.

We will say that a commitment is *well-defined*, if after the commit phase, there is at most one value that can be opened successfully.

LEMMA 1 *The above bit commitment scheme $[[b]]_{C,V}$ is secure against an adaptive adversary: If C remains honest during the commit phase, then the adversary gets at most a negligible amount of information on b, and furthermore, the probability that the opening phase fails is at most 2^{-k}. Moreover, the event that V remains honest throughout and the commitment is not well-defined has probability at most 2^{-k}.*

The first claim follows from the fact that clearly, the adversary has at most a negligible amount of information about $\mathbf{a}$ as a result of the generation of the commitment $[\mathbf{a}]$, and $\mathbf{z}$ is an "encryption" of $\mathbf{v}_b$ with the "one-time pad" $\mathbf{a}$. The claim about the opening phase follows immediately, since $e_0 \leq 2^{-k}$.

To show the second, note first the adversary loses immediately if he corrupts V, so we may assume that this does not happen. Then the intuitive idea is that although C may open $[\mathbf{a}]$ in many different ways (since it is only statically secure), the maximum number is at most 2^n (except with negligible probability). This is a negligible fraction of the possible 2^{2n+k+1} strings. This means that right after having made $[\mathbf{a}]$, C is effectively committed (in the *adaptive* sense) to a negligible size subset of the possible strings[7].

We argue as follows. Pretend for the moment that the IC-signatures offer perfect security (more precisely, $e_1 = 0$). Then consider the set B of players who are still honest after $[\mathbf{a}]$ is opened. These players have of course not changed their minds about the shares they received initially. So they either have consistent shares determining some string $\mathbf{a}'$ which implies that the dealer can only have opened successfully as $\mathbf{a}'$. Or they have inconsistent shares, meaning that C will be deemed corrupt. So it follows that for each set B there is at most one string that the dealer can claim successfully. And there are at most 2^n such sets.

Since $e_1 \leq 2^{-k-1}$, it follows that, except with probability at most 2^{-k-1}, C is effectively committed to a subset W of strings of size at most 2^n.

To be able to open $[[b]]_{C,V}$ in two different ways in this case, there must exist $\mathbf{a}_0, \mathbf{a}_1 \in W$ such that $\mathbf{z} = \mathbf{v}_i \oplus \mathbf{a}_i$, $i = 0, 1$. It follows that $\mathbf{v}_0 \oplus \mathbf{v}_1 = \mathbf{a}_0 \oplus \mathbf{a}_1$. Note that $\mathbf{v}_0 \oplus \mathbf{v}_1$ is a uniformly random $2n+k+1$ bit string (different from the all-zero string $\mathbf{0}$), and is independently distributed from W. But the number of different $\mathbf{a}_0 \oplus \mathbf{a}_1 \neq \mathbf{0}$ with $\mathbf{a}_0, \mathbf{a}_1 \in W$ is clearly at most $2^{2n} - 2^n$. Thus we have that the probability that C can

[5] If the original scheme allows for q different values, then running it l times in parallel, yields a statically secure commitment scheme that allows for q^l different values

[6] This does not depend on whether the adversary is adaptive or not

[7] Elements of our proof are reminiscent of a method introduced by M. Naor [14] in the context of ordinary, computationally secure commitments from pseudo-randomness

open the commitment in two different ways is at most $(2^{2n} - 2^n)/(2^{2n+k+1} - 1) \leq 2^{-k-1}$.

We conclude that the overall probability of cheating is at most $2^{-k-1} + 2^{-k-1} = 2^{-k}$.

4 VSS from Commitments

In [12], a general techniques is presented, given any commitment scheme, for giving zero-knowledge proofs on committed bits, i.e. a prover can commit to a set of bits, and convince a verifier in zero-knowledge that the committed bits satisfy any predicate that can be computed in polynomial time. At the heart of this is a technique (attributed there to Rudich) for building from any commitment scheme a new one, where comparison of committed bits is possible, i.e. the prover can convince the verifier about the XOR of two committed bits, without revealing further information. In the following, we apply this construction to the scheme from the previous section, and we denote a commitment to b in this new scheme by $[[b]]^X_{C,V}$, where again C is the committer and V is the player than can verify relations on committed bits. Since the construction treats the incoming bit commitment scheme as a black box and does not use any assumptions about the way it works, the adaptive security of commitments is not affected by this.

Given such a tool, it is clear that a dealer in a VSS protocol can use the commitment scheme we just developed to commit to all inputs and outputs of a run of the $Distr$ algorithm in our SS scheme, and prove in zero-knowledge to the rest of the players that indeed the committed inputs result in the committed outputs (the shares of some secret). This almost immediately leads to a VSS protocol. However, apart from the fact that this may result in a huge loss of efficiency compared to the underlying SS scheme, it would also give a protocol whose actions depends heavily on which particular secret sharing scheme is used. Below, we give a protocol achieving something slightly stronger, namely a VSS protocol that works given only black-box access to a secret sharing scheme and furthermore does not rely on any particular properties of this scheme. As we shall see, this matches the impossibility result we prove about MPC later.

We begin by a single verifier VSS protocol, i.e. where a single player V can verify the actions of the dealer D.

As a tool for this, we need a protocol that produces from a commitment $[[x]]^X_{D,V}$ for any x a new commitment $[[x]]^X_{P_j,V}$, where P_j can be any player. This must reveal no information to the adversary if both D and P_j remain honest; and even if they are both corrupt, it must still be guaranteed that the two commitments contain the same value. This is what is called a Commitment Transfer Protocol (CTP) in [6]. We generalize the idea from there to build such a protocol for our scenario:

CPT Protocol

1. D sends x and all random inputs used in creating $[[x]]^X_{D,V}$ privately to P_j. Thus, if D is honest, P_j is now in a position equivalent to having created $[[x]]^X_{D,V}$ himself.
2. P_j now creates $[[x]]^X_{P_j,V}$, and proves, using the general techniques described above, in zero-knowledge that it contains the same value as $[[x]]^X_{D,V}$, acting as if he created it himself using the data received in the previous step. If this proof succeeds, the protocol ends here accepting $[[x]]^X_{P_j,V}$.
3. If the proof fails, it is clear that at least one of D, P_j is corrupt. D must then open $[[x]]^X_{D,V}$ in public, and P_j is assigned a default commitment to the opened value x.

With this, we can build our single verifier VSS protocol:

Single Verifier VSS

1. D makes a commitment $[[b]]^X_{D,V}$, where b is the bit D wants to VSS.
2. D chooses random bits $r_1, ..., r_{2k}$ and, for each i generates shares $s_{i1}, \ldots, s_{in}$ where r_i is the secret. Next, D makes commitments to all these values, resulting in commitments $[[r_i]]^X_{D,V}, [[s_{ij}]]^X_{D,V}$.
3. V chooses at random a subset E consisting of half the indices $1, \ldots, 2k$, and broadcasts it. Now, for each $i \in E$, D must open all commitments to r_i and each s_{ij}, and all random inputs used to generate those shares s_{ij}.

4. V verifies that all openings of commitments were valid, and for each $i \in E$, that each set of shares s_{ij} consistently determines r_i (he does it by running the distribution algorithm with the random input broadcasted in the previous step). All information with $i \in E$ can now be discarded.

5. Write $\overline{E}$ for $\{1, \ldots, 2k\} \setminus E$. For each $i \in \overline{E}$, D computes $c_i = b \oplus r_i$, broadcasts it, and using the general techniques described above, D convinces V in zero-knowledge that the XOR of the contents of $[[b]]^X_{D,V}$ and $[[r_i]]^X_{D,V}$ is c_i indeed.

6. Finally, for each $i \in \overline{E}$ and each j, a CPT protocol is executed to convert $[[s_{ij}]]^X_{D,V}$ to $[[s_{ij}]]^X_{P_j,V}$.

In order to ensure that some value for the secret is (almost) always defined after the distribution, we adopt the convention that if V rejects in the above protocol, default consistent shares are assigned to the players.

To later open such a VSS, the following is done:

Single Verifier VSS reconstruct

1. For each $i \in \overline{E}$, each player P_j opens his commitments to all the s_{ij}'s he knows. All the data are sent to V.

2. V verifies all openings and discards those s_{ij}'s for which the commitment was not correctly opened. For those i's where a qualified set of shares (supposedly of r_i) remains, he reconstructs a value r'_i.

3. V XOR's r'_i with the value for c_i broadcasted earlier. This gives a set of bits (which will all be equal to b if D has been honest). V decides by majority among these bits the final value to reconstruct.

We now argue that this single verifier VSS works: first, if D remains honest, it follows directly from the security of the commitment scheme that the adversary gets (almost) no information on b.

On the other hand, suppose V remains honest. The intuition then is that due to the cut-and-choose, a majority of the r_i with $i \in \overline{E}$ are correctly shared, with the players committed to their shares. Hence, these r_i can be reconstructed, and when V XOR's these with the corresponding c_i's, V gets the bit b as a result in a majority of the cases. A more detailed analysis follows.

Consider for some $i \in \overline{E}$ the commitments $[[r_i]]^X_{D,V}$ and the committed shares $[[s_{ij}]]^X_{D,V}$. Call the index i *good* if the s_{ij} indeed consistently determine r_i and call it *bad* otherwise.

Now if at least a constant fraction of the indices, say $0.1k$, are bad then except with negligible probability, V will reject in Step 4 of the distribute protocol (in particular, the subset E chosen by V will contain at least one bad index). On the other hand, if at most $0.1k$ indices are bad, then even if all bad indices are outside E, the good ones will be in majority among the ones used in the reconstruction phase, and the correct value of b will be computed.

VSS Protocol

To finally prove Theorem 1, just note that a dealer can perform n single verifier VSS's on the same bit b where every player gets to play the role of V. As a side effect of this, we get n commitments $[[b_i]]^X_{D,P_i}$, where if D has been honest $b_1 = \ldots = b_n = b$. So clearly, all we need is for D to convince separately every player that this relation holds. The general zero-knowledge techniques mentioned earlier will suffice for this.

5 Multiparty Computation

In this section we prove our impossibility result, Theorem 2.

Before doing so, we first point out, as claimed earlier, that secure computation of linear functionals can be efficiently handled using black-box SS, both in the passive and active models. As a consequence, Theorem 2 can also be interpreted as an impossibility result essentially regarding secure multiplication, or equivalently, Oblivious Transfer.

In the passive case, this is trivial: each input bit b is split randomly into $b = b_1 \oplus b_2 \oplus \ldots \oplus b_n$, and b_i is given to player P_i. Each player then computes the desired linear function locally on the b_i's and publish the result. The global result is then the xor of the local results. Note that the black-box SS-scheme is not needed for this. In the active model, we can first establish a situation where the input bits and the b_i's are verifiably secret shared. The players then prove using general techniques that they performed their local computations correctly (see Section 4).

To prove Theorem 2, let us first recall the standard argument showing the impossibility result when no SS-oracle is given. As mentioned, a $Q2$ adversary structure X is called *maximal* if there does not exist a $Q2$ adversary structure $X' \neq X$ such that $X \subset X'$. For the sake of contradiction suppose that for every maximal adversary structure $\mathcal{A}$ there exists a protocol which runs in time bounded by some polynomial, and which computes f_{AND} securely against $\mathcal{A}$. Since the number of maximal $Q2$ adversary structures is double-exponential and the number of polynomial-time protocols is single exponential then (by a counting argument) there must exist a protocol π computing f_{AND} securely against two different maximal $Q2$ adversary structures X and Y. This means that π is secure against $Z = X \cup Y$. By maximality of X and Y we have that Z is not $Q2$. Therefore f_{AND} cannot be computed securely against it, and we have a contradiction.

In our case the situation is more difficult because the behavior of the players may depend on the oracle answers. Observe that when the SS-oracle is asked by a set of players A to reconstruct some secret then the protocol gets the information whether A is a member of the adversary structure. Thus we may assume that together with reconstruction request comes a query about a membership in the adversary structure and that together with the SS-oracle we have a membership oracle.

Therefore for two different adversary structures X and Y the same protocol may behave in two different ways if it happens to ask a membership query about a set in a symmetric difference of X and Y. Intuitively the biggest combinatorial difficulty of the proof is to show that there always exist two different maximal $Q2$ adversary structures A_S and A_T and a protocol δ working against both of them, such that δ will, with large probability, not ask a membership query about any set in a symmetric difference of X and Y.

More precisely, the proof proceeds as follows. It is enough to prove that for any polynomial $p()$ the collection of oracle protocols of size $p(n)$ cannot handle all maximal $Q2$ adversary structures on n players. So for the sake of contradiction suppose that there is a polynomial $p()$ such that for every set of players P of size n and every maximal $Q2$ adversary structure $\mathcal{A} \subseteq \mathcal{P}(P)$ there exists an SS-oracle protocol $\pi(\mathcal{A})$ of size at most $p(n)$ computing f_{AND} securely against $\mathcal{A}$. All such protocols can be specified by a polynomial number of bits, and hence the total number of such protocols is at most a single exponential in n.

We will then show:

LEMMA 2 *For every n large enough there exist two adversary structures $A_S, A_T \subseteq \mathcal{P}(P_n)$ such that*

1. *the size of the set of players P_n is $2n + 2$,*
2. *$\pi(A_S) = \pi(A_T)$, and*
3. *$\pi(A_S)$ asks a membership query about a set in the symmetric difference of A_T and A_S with probability at most $2^{-1.5n}$. This probability is taken over all random choices made in the protocol, and over a random choice of the $2n + 2$ input bits. Moreover, A_S contains a set A, and A_T a set B, such that $|A| = |B| = n + 1$ and $A \cup B = P_n$*

Before proving this, let us show how the existence of such A_T and A_S yields the contradiction. We will construct from the protocol $\pi_0 := \pi(A_S) = \pi(A_T)$ a new protocol for two players, Alice and Bob, with input bits b_A, b_B that will compute securely (and with negligible error probability) $b_A \wedge b_B$. This is well known to be impossible, even if only passive cheating occurs, and the honest Alice and Bob are allowed unbounded computing power.

Consider the sets $A \in A_S$ and $B \in A_T$ guaranteed by the lemma. We let Alice and Bob simulate an execution of π_0, where Alice controls the players in A and Bob those in B. Alice selects as input bits for players in A a random set of $n + 1$ bits such that the AND of all of them equals b_A. Similarly for Bob. Then we execute π_0, where Alice (Bob) executes the algorithms of players in A (B). Every message from a player in A to a player in B causes Alice to send the message to Bob, and vice versa.

Note that although efficiency of protocols plays a crucial role in proving the above lemma, we do not need to be concerned about efficiency at this point anymore, because we are now headed towards establishing a contradiction by building a protocol for a problem for which no protocol exists, even if unbounded computing is allowed. Hence we need no SS-oracle, we can use an arbitrary (inefficient) secret sharing scheme for $A_S \cup A_T$. Also, we may assume that Alice and Bob each have a list of the sets in A_T and A_S. They will use it to answer membership queries as follows: in most cases π_0 asks about a set which is in both A_S, A_T or in neither of them, so it is clear what the answer should be. In the unlikely event that the question is about a set in the symmetric difference, the protocol stops, we say it crashes. Alice and Bob use the result computed by π_0 as their output if the protocol finishes; if it crashes they let the output but be 1.

Observe that in case $b_A = 0$ or $b_B = 0$, the probability of a crash is at most $2^{-n/2+2}$: if, say, $b_A = 0$, we choose randomly between a set of at least 2^n inputs, namely all those inputs to players in A, where at least one bit is 0. These cases constitute at least a fraction 2^{-n-2} of the overall probability space, so even restricted to this case, the crash probability is at most $2^{-1.5k}/2^{-n-2} = 2^{-n/2+2}$. This immediately implies that Alice and Bob compute correctly $b_A \wedge b_B$ except with negligible probability in n. It also implies that privacy is satisfied: it is enough to argue that if $b_A = 0$, Alice learns a negligible amount of information about b_B. To see this, consider an idealized scenario, where there are no crashes and all membership queries are answered according to A_S. Then since π_0 is secure against A_S, it follows that whenever the players in A (alias Alice) have 1 or more zeros in their input, they learn almost no information about the inputs of players in B. However, the only difference between the actual protocol we specified for Alice and Bob and the idealized case is the crashes. And since crashes occur with negligible probability, it follows that Alice's view of the actual protocol is statistically indistinguishable from what she sees in the idealized case.

This completes the proof that Alice and Bob would be able to compute the AND function securely, and so we have our contradiction.

Let us now show the existence of A_S and A_T satisfying the conditions (1)–(3).

It will be enough to restrict ourselves to a certain class of maximal $Q2$ structures. For a given n we will construct a class $\mathcal{C}_n$ of $2^{2^{1.9n}}$ such structures as follows. Take a set of players P_n such that $|P_n| = 2(n+1)$. Define *split* to be a pair $(X, P \setminus X)$ such that $|X| = |P \setminus X| = n+1$. Fix in an arbitrary way a set of splits SP_n that has a property that $(X, Y) \in SP_n$ if and only if $(Y, X) \notin SP_n$ (for example: fix a player $p_0 \in P_n$ and define SP_n to be a set of all splits $(X, P \setminus X)$ such that $p_0 \in X$).

For a technical reason we make a further restriction, and choose an arbitrary subset R_n of SP_n of a size $2^{1.9n}$ (we have $|SP_n|$ is $\Omega(2^{2n}/\sqrt{n})$ by standard combinatorics, therefore this operation is always possible).

Now observe that every subset $S \subseteq R_n$ determines a unique adversary structure A_S in the following way: a set of players $Z \subseteq P_n$ belongs to A_S if and only if one of the following conditions is satisfied:

- $|Z| < n+1$,
- $(Z, P \setminus Z) \in S$, or
- $(P \setminus Z, Z) \notin S$.

Such an adversary structure will be called a *split structure*. We define $\mathcal{C}_n$ to be the set of all split structures A_S (where $S \subseteq R_n$). Clearly every split structure is a maximal $Q2$ structure.

To avoid to many subscripts we fix n. From now on we will consider only protocols running against the split structures (what we can safely assume because we are proving a negative result). Let $prot$ be the set of all the protocols assigned to the set of all split structures by π (i.e. $prot = \pi(\mathcal{C}_n)$). It is easy to see that now all the membership queries about the sets of a size at smaller than $n+1$ are always answered positively. Similarly all queries about the sets of a size bigger than $n+1$ are always answered negatively. The only queries which give some information about the adversary structure are the queries about the sets of the size exactly $n+1$. Therefore we can now assume that instead of a membership oracle for A_S every protocol is given a membership oracle for S. This assumption simplifies a bit the notation and views the problem in a more abstract way.

Let t be the maximum of the expected number of queries asked by the protocols from the set $prot$ (more precisely let $t = \max_{A \in \mathcal{C}_n}$ (expected number of queries asked by $\pi(A)$ when it runs against the structure A)). Let $s = 2^{1.7n}$ (the choice is somewhat arbitrary, what matters is that s is much bigger than t, but much smaller that $|R_n|$.

Divide R_n into s blocks of equal size in an arbitrary way (this operation will always be possible for big enough n). Let $B_1, \ldots, B_s$ be the resulting blocks. We will say that a set X is *blinking in a block B_j* iff there exists a set Y such that all the following conditions are satisfied:

- the protocols assigned by π to X and Y are the same, i.e. $\pi(X) = \pi(Y)$

- $X \cap B_j \neq Y \cap B_j$, and
- $X \cap (R_n \setminus B_j) = Y \cap (R_n \setminus B_j)$.

The last two conditions mean in other words that X and Y differ on a set B_j and do not differ elsewhere. The intuition here is that the protocol $\pi(X)$ may have some difficulty in deciding if it is running against X or Y, since it must ask a membership query in B_j to find out.

LEMMA 3 *For every big enough n there exists a set blinking everywhere (i.e. there exists $S \subseteq R_n$ such that S is blinking for every block $B_1, \ldots, B_s$).*

PROOF. For every block B_j let N_j denote the family of sets *not* blinking for B_j. What we need to show is that $\cup_{j=1}^{s} N_j \neq \mathcal{P}(R_n)$. We will actually prove a stronger fact, namely

$$\sum_{j=1}^{s} |N_j| < 2^{2^{1.9n}}. \tag{1}$$

Fix an arbitrary B_j. Take an arbitrary set $Z \subset R_n \setminus B_j$. Now take the family $\mathcal{G} = \{ W \subseteq R_n : W \setminus B_j = Z \}$ (in other words $\mathcal{G}$ is a family of all sets whose projection on $R_n \setminus B_j$ is equal to Z). If two different sets in $\mathcal{G}$ were assigned the same protocol, then they would both blink in B_j, so it follows that the size of the family of sets in $\mathcal{G}$ that are not blinking in B_j cannot be bigger than the number $|prot|$ of different protocols. Therefore after summing over all possible sets $Z \subset R_n$ we have that $|N_j| \leq |prot| 2^{2^{1.9n}(s-1)/s}$. Since the choice of B_j was arbitrary we get that the left-hand-side of (1) is not bigger than $s|prot|2^{2^{1.9n}(s-1)/s}$. Therefore to prove (1) is enough to show that

$$s|prot|2^{2^{1.9n}(s-1)/s} < 2^{2^{1.9n}}$$

which is equivalent to

$$s|prot| < 2^{\frac{2^{1.9n}}{s}} = 2^{2^{0.2n}} \tag{2}$$

The left hand side of (2) is single exponential in n and so for big enough n the inequality (2) (and hence (1)) holds. $\triangle$

Let now n be big enough that the blinking everywhere set exists. Let $S \in R_n$ be such a set. Thus for every block B_j there exists a set $blink(B_j) \in \mathcal{C}_j$ such that

- $\pi(A_S) = \pi(blink(B_j))$,
- $S \cap B_j \neq blink(B_j) \cap B_j$, and
- $S \setminus B_j = blink(B_j) \setminus B_j$.

Consider the runs of $\pi(A_S)$ against the adversary structure A_S and consider the probability distribution of these runs over a random choice of the input bits to the computation as well as the random coins used. For every B_j let $pr(B_j)$ be the probability that $\pi(S)$ asks a query about some element in B_j. It is easy to see that $\sum_j pr(B_j)$ is the expected number of queries asked by $\pi(A_S)$. Recall that this expected number of queries is polynomial in n, and hence it is, for all large enough n, smaller than s by a factor of at least $2^{1.5n}$. Therefore

$$\sum_{j=1}^{s} pr(B_j) < \frac{s}{2^{1.5n}}$$

Thus the average value of $pr(B_j)$ is at most $2^{-1.5n}$. Let B_l be such that $pr(B_l) \leq 2^{-1.5n}$. In other words, with probability at least $1 - 2^{-1.5n}$ the machine $\pi(A_S)$ will never ask about any element in B_l. Therefore if we set $T = blink(B_l)$ then the protocol $\pi(A_S)$ (which is by the way equal to $\pi(A_T)$) with a probability $1 - 2^{-1.5n}$ will not distinguish between A_S and A_T.

6 Error Free Protocols and Open Problems

In this paper, we have dealt with the situation where a broadcast channel (in addition to the private ones) is available and access structures are $Q2$. It is known [10] that if the adversary structure is $Q3$ (no three sets in the adversary structure covers the player set) and no broadcast is given, then VSS and MPC with zero error probability is possible. Thus it is natural to ask if in this model we are given an *error free* SS scheme, can we build an *error free* VSS scheme with polynomially related efficiency?

We sketch here how to build an error-free commitment scheme. The construction requires a broadcast channel, however, such a channel can be simulated, given an efficient way to decide membership in the adversary structure (see [7]), and the secret sharing scheme we assume gives precisely such a decision procedure.

The commitment scheme works as follows: the committer shares his secret s to get shares $s_1, ..s_n$. He further shares each s_i to get sets of subshares $\{s_{ij}\}$. He sends $s_{1j}, ..., s_{nj}$ to P_j and sends s_i plus the random bits used in sharing s_i to P_i. This allows both P_i and P_j to compute s_{ij}, so they can privately compare their values and ask the committer to publicly announce s_{ij} if there is a mismatch. If some P_i realizes that the committer is corrupt, P_i accuses the committer, who must then make public all data sent to P_i.

In the same way as in the VSS from [2], this will ensure that the committer is always either disqualified because there are too many complaints, or the honest players agree on all subshares they know.

To open the commitment, the committer will make s and the shares $\{s_i\}$ public. Also each P_i will make his share s_i and the subshares $\{s_{ij}\}$ public. Each P_j announces if he agrees with s_{ij} or not. The share announced by P_i approved if all players except a non-qualified subset agree with the subshares he claims. The opening is accepted, if and only if the shares claimed by the committer are consistent and agree with all approved shares announced by the other players.

Since all players who have remained honest agree on subshare values, all honest players will always have their shares approved. Therefore, if the honest players have inconsistent shares, the commitment cannot be opened convincingly. But if the honest players do have consistent shares, the $Q3$ property implies that the commitment can only be opened in one way. On the other hand, if the dealer remains honest, then the honest players will have consistent subshares of s_i, even if P_i is corrupt. Hence P_i can (again by the $Q3$ property) only have the correct s_i approved, so an honest committer can always open successfully.

Naturally, this commitment scheme can be used to build a VSS scheme based on zero-knowledge techniques as shown earlier. But this scheme will have a non-zero error probability. We do not know how to build error free VSS efficiently from error free SS for this scenario, and leave this as an open problem.

References

[1] A. Beimel: *Secure Schemes for Secret Sharing and Key Distribution*, Ph.D.-thesis, Technion, Haifa, June 1996.

[2] M. Ben-Or, S. Goldwasser, A. Wigderson: *Completeness theorems for Non-Cryptographic Fault-Tolerant Distributed Computation*, Proc. of ACM STOC '88, pp. 1–10, ACM Press, 1988.

[3] D. Chaum, C. Crépeau, I. Damgård: *Multi-Party Unconditionally Secure Protocols*, Proc. of ACM STOC '88, pp. 11–19, ACM Press, 1988.

[4] B. Chor, S. Goldwasser, S. Micali, B. Awerbuch: *Verifiable Secret Sharing and Achieving Simultaneity in the Presence of Faults*, Proc. of IEEE FOCS '85, pp. 383–395, 1985.

[5] R. Cramer, I. Damgård, S. Dziembowski, M. Hirt, T. Rabin: *Efficient Multiparty Computations with Dishonest Minority*, Proc. of EUROCRYPT '99, Springer Verlag LNCS, vol. 1592, pp. 311–326, 1999.

[6] R. Cramer, I. Damgård, U. Maurer: *Secure and Efficient General Multiparty Computations from any Linear Secret Sharing Scheme*, Proc. of EUROCRYPT '00, Springer Verlag LNCS, 2000.

[7] M. Fitzi, U. Maurer: *Efficient Byzantine Agreement Secure against General Adversaries*, Proc. of Distributed Computing (DISC '98), Springer Verlag LNCS, vol. 1499, pp. 134–148, 1998.

[8] R. Gennaro: *Theory and Practice of Verifiable Secret Sharing*, Ph.D.-thesis, Massachusetts Institute of Technology, 1995.

[9] O. Goldreich, S. Micali and A. Wigderson: *How to Play Any Mental Game or a Completeness Theorem for Protocols with Honest Majority*, Proc. of ACM STOC '87, pp. 218–229.

[10] M. Hirt, U. Maurer: *Player Simulation and General Adversary Structures in Perfect Multiparty Computation*, Journal of Cryptology, 13(1):31–61, 2000. Extended abstract in Proc. of ACM PODC'97, pp. 25–34, ACM Press, 1997.

[11] M. Ito, A. Saito, T. Nishizeki: *Secret sharing schemes realizing general access structures*, Proc. of IEEE GlobeCom '87 Tokyo, pp. 99-102, 1987.

[12] J. Kilian: *A note on Efficient Proofs and Arguments*, Proc. of ACM STOC '92, pp. 723–732, ACM Press, 1992.

[13] M. Karchmer, A. Wigderson: *On Span Programs*, Proc. of Structure in Complexity '93, pp. 102–111, 1993.

[14] M. Naor: *Bit commitment using pseudorandomness*, Journal of Cryptology, 4(2):151-158, 1991.

[15] T. Rabin, M. Ben-Or: *Verifiable Secret Sharing and Multiparty Protocols with Honest majority*, Proc. of ACM STOC '89, pp. 73–85, ACM Press, 1989.

[16] A. Shamir: *How to Share a Secret*, Communications of the ACM 22 (1979) 612–613.

[17] A. Yao: *Protocols for Secure Computation*, Proc. of IEEE FOCS '82, pp. 160–164, 1982.