Unconditionally Non-interactive Verifiable Secret Sharing Secure against Faulty Majorities in the Commodity Based Model

Anderson C.A. Nascimento[1], Joern Mueller-Quade[2], Akira Otsuka[1], Goichiro Hanaoka[1], and Hideki Imai[1]

[1] Institute of Industrial Science, The University of Tokyo
4-6-1, Komaba, Meguro-ku, Tokyo, 153-8505 Japan
{anderson,otsuka,hanaoka,imai}@imailab.iis.u-tokyo.ac.jp
[2] Universitaet Karlsruhe, Institut fuer Algorithmen und Kognitive Systeme
Am Fasanengarten 5, 76128 Karlsruhe, Germany
muellerq@ira.uka.de

Abstract. This paper presents a non-interactive verifiable secret sharing scheme (VSS) tolerating a dishonest majority based on data predistributed by a trusted authority. As an application of this VSS scheme we present very efficient unconditionally secure multiparty protocols based on predistributed data which generalize two-party computations based on linear predistributed bit commitments. The main results of this paper are a non-interactive VSS where the amount of data which needs to be predistributed to each player depends on the number of tolerable cheaters only, a simplified multiplication protocol for shared values based on predistributed random products, a protocol for fair exchange of secrets based on predistributed data, and non-interactive zero knowledge proofs for arbitrary polynomial relations.

Keywords: Verifiable secret sharing, pre-distributed data, multiparty protocols.

1 Introduction

This paper gives a protocol for information theoretically secure verifiable secret sharing (VSS) in the commodity based model which tolerates a dishonest majority. On the basis of pre-distributed data a dealer can share a secret such that all players are convinced that the shares they hold are valid, i. e., sets of players larger than a threshold t can reconstruct the shared secret. When dealing with an adversary which is able to corrupt a majority of the players, this requirement is slightly relaxed, since corrupted players are always able to abort the protocol.

As an application of this VSS scheme we present very efficient multiparty protocols in the commodity based model which can tolerate up to $t < n$ corrupted parties. If the number t is known the protocols can be chosen to be robust against $n-t$ players trying to abort the calculation. The protocols can be seen as a generalization of [21] to multiparty protocols. But the multiplication procedure used here is much simpler than the one in in [21] and compared to trivial extensions of [21] like pre-distributing two-party computations like oblivious transfer and then applying a general construction like [13] we save data. The amount of data to be pre-distributed to one party depends, for fixed security parameter and fixed size of the field, only on t and not on the total number n of parties. Due to the advantages of the commodity based model we can obtain a non-interactive VSS allowing non-interactive zero knowledge proofs.

VSS. Tompa and Wolf [29] and McEliece and Swarte [20] were the first to study the problem of secret sharing in presence of a corrupted majority. They proposed solutions which work when the dealer is honest and corrupted players may attempt to cheat during the reconstruction of the secret. Later, Chor et al. [7] defined a complete notion of VSS, and gave a solution which was based on some intractability assumptions.

In Ben-Or et al. [4], an information theoretically secure VSS scheme was proposed which worked against any adversary which corrupts up to less than 1/3 of the players and the dealer. In [4], it was assumed that the players are connected by pairwise secure channels (the so-called secure channels model). As VSS, when implemented without a broadcast channel, implies Byzantine Agreement, the results of [17] show that the solution of [4] is optimal in the secure channels model.

Rabin and Ben-Or [23] were able to show that in a secure channels plus broadcast channel model unconditionally secure VSS is possible against any dishonest minority. In [11], Cramer et al. proposed a VSS scheme secure against an adaptive adversary which can corrupt any dishonest minority by using a linear information checking protocol. The protocols of [11] and [23] are interactive.

While secrecy in the secure channels plus broadcast channel model can easily be maintained even against an adversary which corrupts a majority of players, the same cannot be said of the validity of the shares. Clearly, correct verifiability of the shares in the presence of a faulty majority cannot be achieved in the secure channels plus broadcast channel model without further assumptions [23].

Assuming that the discrete logarithm problem is intractable Feldman proposed a VSS scheme [12] where the verifiability of the shares is information theoretically secure but the secrecy of the secret is only computationally secure. Pedersen [22] proposed a “dual” of Feldman’s scheme. Pedersen’s scheme protects the secrecy of the secret unconditionally, while verifiability is protected only computationally under the assumption that the discrete logarithm problem is intractable.

In this paper, we introduce a VSS scheme based on pre-distributed data which is information theoretically secure against *dishonest majorities*, that is, both *the secrecy of the secret* and the *verifiability of the shares* are achieved independently of how much computational power is available to an adversary. Moreover, our solution is non-interactive.

Multiparty Computation. As an application of our protocol, we provide a very simple and efficient information theoretically secure multiparty computation protocol based on pre-distributed data which is secure against dishonest majorities. In [1,2,10] protocols for multiparty computations secure against a faulty majority were proposed. In these papers, it was assumed that all the players were connected by oblivious transfer channels and a broadcast channel was available. Due to the use of pre-distributed data instead of oblivious transfer the protocols presented here are more efficient and do not need zero-knowledge proofs based on cut-and-choose arguments which increase the round complexity of protocols. Furthermore our protocol performs computations directly over a field $GF(q)$ and not only over binary fields.

Finally, it should be remarked that, if a protocol is secure against *any* dishonest majority, even a single player should be able to abort the computation as was pointed out in [10]. However, the protocols for secure computation proposed in [1,2,10,13] can be aborted by a single player *even when the number of honest players is known to be much larger than one.* This is not the case with our protocol. Given that there are n players of which at most t are dishonest, then there exists a secure VSS protocol for which $n - t$ players are necessary to abort the execution of this protocol. This property also holds for the application to multyparty protocols.

1.1 Commodity Based Cryptography and Related Work

In [3] the commodity based cryptographic model was introduced on which the protocols presented here are based. In this model players buy cryptographic primitives from “off-line” servers. These primitives can be used later on to implement general cryptographic protocols. The commodity based model was inspired in the internet architecture, which is usually based on the “client-server” paradigm. Once the primitives, or *commodities* as they are called by Beaver, are acquired, no further interaction between the server and the users is required. Therefore, the server need not know any secret values of the players.

In this contribution, we show that the use of off-lines servers provides very efficient and simple protocols for verifiable secret sharing and secure function evaluation over $GF(q)$ in the presence of a faulty majority.

Although this model was formalized just in [3], several independent works share the same flavor. We cite key-pre-distribution schemes [18], unconditionally secure bit commitments [24,6] and unconditionally secure digital signature schemes [16].

The work which comes closest to the application of our VSS scheme to multiparty computations is [21]. There secure protocols for two-party computations in the commodity based model are proposed. Our protocol for multiparty secure computation can be understood as an extension of [21].

1.2 Our Contribution

In this Section we summarize our contribution. Due to the assumption that there is a trusted center which pre-distributes data during a setup phase, we could design a protocol for verifiable secret sharing which has the following interesting features.

– It is the first VSS protocol where the security of the secret and of the verifiability are achieved even against an all-mighty adaptive adversary which can corrupt any majority of the players and the dealer.
– It is non-interactive
– The verifiability of the protocol is not based on any cut-and-choose argument or expensive zero knowledge proofs.
– It is conceptually very simple.

Furthermore for a fixed security parameter k and a fixed field size q the amount of data which has to be pre-distributed depends on t only.

As an application of our VSS, we propose a protocol for secure multiparty computations which also shows very interesting features (which to the best of our knowledge for the first time appear together in a single protocol): It is based on novel verifiable primitives in the commodity based model which allow two players to perform secure multiplication of shares over $GF(q)$; It is information theoretically secure against any adversary which can corrupt any majority of the players; and given that there are n players of which at most t are dishonest, $n-t$ players are necessary to abort an execution of the multy party computation protocols.

2 Model

Here, we present a model for *non-interactive verifiable secret sharing protocol based on pre-distributed data.* Note that we do not give a general definition of a verifiable secret sharing scheme, but rather propose a model we believe is general enough to cover any non-interactive protocol based on pre-distributed data. For a general definition of a verifiable secret sharing scheme, please look at [23]. We work in the commodity based cryptography model as proposed by Beaver in [3]. There are n players $\{P_1, P_2, \dots, P_n\}$ a dealer D and a trusted center. Also, we assume the existence of an authenticated broadcast channel. Note that this assumptions are only made to simplify the protocol presentation as a broadcast channel can as well be pre-distributed by the trusted center during a setup phase [3]. The trusted center is supposed to act only during a setup phase and no sensitive information concerning the players input is ever transmitted to it. The players are connected to the trusted center by secure channels. Secure channels between the parties are assumed, too, but can be pre-distributed as well.

We assume a central adversary with unbounded computational power who actively corrupts t players, $t < n$.

Notation. For a VSS scheme Π with a trusted center T, a set of dealers $\mathcal{D}$, and a finite set of players $\mathcal{P}$ the protocol consists of ($\mathcal{S}$, $\mathcal{K}$, $\mathcal{V}$, Commit, Share, Verify), where

- $\mathcal{S}$ *is a finite set of possible secrets,*
- $\mathcal{K}$ *is a finite set of possible signing-keys,*
- $\mathcal{V}$ *is a finite set of possible verification-keys,*
- $\mathcal{C}$ *is a finite set of possible commitments,*
- $\mathcal{S}_h$ *is a finite set of possible shares,*
- Commit : $\mathcal{S} \times \mathcal{K} \to \mathcal{C}$ *is a commitment-algorithm,*
- Share : $\mathcal{C} \times \mathcal{P} \to \mathcal{S}_h$ *is a share-generation-algorithm, and*
- Verify : $\mathcal{C} \times \mathcal{S}_h \times \mathcal{V} \to \{\text{accept}, \text{reject}\}$ *is a verification-algorithm.*
- Reconstruct : $\mathcal{S}_h^{t+1} \to \mathcal{S}$ *is a reconstruction algorithm which regains a secret* s *from* $t+1$ *valid shares.*

System Setup by *T*. *For a dealer* $D \in \mathcal{D}$*,* T *chooses a signing-key* $K \in \mathcal{K}$*, and for each player* P_i *in* $\mathcal{P}$*,* T *chooses a verification-key* $v_i \in \mathcal{V}$*, then transmits the keys via private channels to* D *and each* P_i*. Each player keeps the keys secret. After delivering the keys,* T *never engages in the protocol again.*

Share. *On input of a secret* $s \in \mathcal{S}$*,* D *broadcasts a commitment* $c \in \mathcal{C}$ *where* $c = \text{Commit}(s, K)$ *to all players. Then, each player* P_i *computes his share* $s_i \in \mathcal{S}_h$ *where* $s_i = \text{Share}(c, P_i)$*. A verification of the validity of the shares will not be necessary in this stage as this can be guaranteed from the pre-distributed data.*

Reconstruct. *All the players* $\{P_1, P_2, \ldots, P_n\}$ *broadcast their shares* $\{s_1, s_2, \ldots, s_n\}$*. Each player* P_i *checks if there exits a subset of players* $\{P_{i_1}, P_{i_2}, \ldots, P_{i_{t+1}}\}$ with $\text{Verify}(c, s_{i_k}, V_i) = $ *accept for all* $k = 1, \ldots, t+1$ *where* $t+1$ *is equal to or greater than the reconstruction threshold. If this is the case,*P_i runs $\text{Reconstruct}(s_{i_1}, \ldots, s_{i_{t+1}})$ and outputs s, otherwise it outputs Δ.

The scheme is said to be *secure* if it satisfies the following properties[11]:

1. *Termination*: If the dealer D is honest then all the honest players complete the Share protocol and if honest players decide to run the Reconstruct protocol after a successful run of the Share protocol they should generate output (possibly Δ indicating that no secret could be reconstructed).
2. *Secrecy*: If the dealer is honest and no honest player has started Reconstruct, them the adversary has no information about the secret s unless the reconstruction threshold was chosen to be smaller than the number of players the adversary can corrupt.
3. *Correctness:* Once the uncorrupted players complete the protocol Share, there is a fixed value $r \in \mathcal{S}$, so that the following requirement holds: If at least $t+1$ players are willing to reconstruct the secret, each uncorrupted player, with high probability, outputs r at the end of Reconstruct. If there are more than $n-t$ Byzantine faults, all honest players output Δ and the protocol terminates. Furthermore if the dealer was uncorrupted $r = s$.

Remark 1. Note that our definition of security is slightly weaker than the definitions of security for VSS protocols with honest majority [11]. In our definition, cheating parties can prevent the completion of the protocol. When dishonest majorities are in question, this situation cannot be avoided. If the protocol is secure against adversaries which corrupt up to t parties, $t > n/2$, always $n - t$ parties will be able to abort the protocol.

3 A VSS Protocol Based on Pre-distributed Data

The protocol Π consists of the following sub-protocols: Setup, Share, Reconstruct, and Verify. In the unconditionally secure protocol below the probability of cheating successfully equals the probability to successfully guess an element from the field over which computations are done. Hence we choose a prime power q and the field $GF(q)$ with q elements depending on the security parameter, e. g., $q = 2^k$.

For simplicity, we assume $\mathcal{S} = \mathcal{C} = GF(q)$, $\mathcal{K} \in GF(q)[x, y]$, $\mathcal{V} = GF(q) \times GF(q)[z]$ as in the following construction.

The basic intuition behind the protocol is simple. The trusted center will share a random value with the players in a way that each player is committed to his share to each other player. When executing the Share algorithm, the dealer changes this random number, to which he is committed by the pre-distributed VSS, into a commitment to his secret.

Setup. A trusted center T randomly chooses bivariate polynomial $f(x, y) \in GF(q)[x, y]$ such that

$$f(x, y) = \sum_{i=0}^{t+1} \sum_{j=0}^{t+1} a_{ij} x^i y^j,$$

where each coefficient a_{ij} is randomly and uniformly chosen from $GF(q)$, and $t + 1$ is the threshold of the secret sharing scheme.

T sends the bivariate polynomial $f(x, y)$ to the dealer D through a private channel. Then T chooses for each player P_ia random verification key (a secret point) $v_i \neq 0$ from $GF(q)$, and sends v_i, $f(v_i, y)$ and $s_i(x) = f(x, i)$ to P_i through a private channel[1]. After delivering these private keys, T does not engage in the rest of the protocol.

For the random "secret" $a = f(0, 0)$ the polynomial $f(v_i, y)$ will later be used by P_i to verify shares of other players and the polynomial $s_i(x)$ is the share of the party P_i for a.

[1] The index i which is the "name" of a participant is here interpreted as a value of $GF(q)$. To have enough different names we need $q \geq n$.

Share. In this stage of the protocol the shares of a random secret a as well as the verification polynomials will be changed to shares and verification polynomials for a specific secret s.

Let

$$g(x, y) = \sum_{i=1}^{t+1} \sum_{j=1}^{t+1} b_{ij} x^i y^j$$

denote a publicly known polynomial for which $g(0, 0) = 0$. On input s for a secret, the dealer D computes a value c satisfying $c = s - a$, where a is computed as: $a = f(0, 0)$

Then, D broadcasts the value c. Next each party P_i calculates the polynomial $g(v_i, y) + c$ and adds it to the polynomial $f(v_i, y)$ which was obtained by P_i in the sharing phase to obtain the new verification function $g(v_i, y) + c + f(v_i, y)$ for the shared secret s. The shares $s'_i(x)$ for the secret s will be computed by adding $g(x, i)$ to $s_i(x) = s(x, i)$ which was obtained by P_i in the setup phase.

For the secret $s = f(0, 0) + g(0, 0) + c$ the verification functions and the shares computed above have the same distribution as if s would have been equal to the value a used in the setup phase.

As $f(x, y)$ is chosen by the trusted center and $g(x, y)$ is publicly known the validity of the shares computed above is evident and need not be verified at this stage.

Reconstruct. It is enough to show how a shared random secret a is reconstructed. The notation will therefore be as in the setup phase. Each player P_i broadcasts his share $s_i(x)$ over the broadcast channel.

Verify

On receiving a share $s_j(x)$ from the player P_j over the broadcast channel, each player P_i checks the share by checking the following equation:

$$ver_{ji} = \left\{ \begin{matrix} accept \text{ if } s_j(v_i) = f(v_i, j) \\ reject \text{ otherwise} \end{matrix} \right\}$$

If $ver_{ji} = reject$, player P_i broadcasts the message *reject*(j). If more $t + 1$ players broadcast *reject*(j), the player P_j is disqualified. It is easy to see that (with high probability) all honest players will obtain the same result in the verification procedure.

If $n - t$ or more players are rejected, all the players output Δ and the protocol terminates.

If less than $n - t$ players are rejected, there will be a set of $t + 1$ valid shares $s_{i_1}(x), \ldots, s_{i_t}(x)$ in possession of each honest player P_i,. Thus, the secret $a = f(0, 0)$ can be reconstructed by Lagrange interpolation from $s_{i_1}(0), \ldots, s_{i_{t+1}}(0)$.

From this construction we will obtain the following result which we state without proof.

Theorem 1. *The above protocol is a secure VSS in the commodity based model.*

Note that in our protocol, for each player P_i the secret "check" information $v_i, 1 \leq i \leq n$ is never released, so it can be distributed only once for several protocols (even with different dealers). The check information can hence be safely reused within a bigger protocol and reduce communication from the trusted center to the players.

Another interesting fact about this VSS scheme is that, given that each player uses the same verification information v_i in all of its executions, it is linear, that is, the sum of two shares of two secrets becomes a verifiable share of the sum of the secrets. We state the following proposition without proof.

Proposition 1. *Denote by $[s]_i$ the pair of the verification function and the share held by a player P_i for a given secret s. Then for two secrets a, b shared with the above VSS scheme using the same verification information v_i for each player P_i and a value $\lambda \in GF(q)$ it holds that $[a]_i + \lambda[b]_i = [a + \lambda b]_i$.*

Note that the commitment algorithm Commit(s, K) is very related to the idea of check vectors. Pre-distributed commitments are very much similar to Rabin's "check vectors" [23]. The idea of a "check vector" is that a party A provides some correlated secrets to parties B and C, and these secrets let B send an authenticated message to C. This is similar to what we do, except that the typical use of check-vectors in the literature, party A knows the message (whereas in our constructions it does not). Thus, as party A need not know the message when it distributes the check vectors, it can be used in an off-line way.

We note that the trusted center can share a random secret with the players. Moreover, the trusted center can pre-distribute shares to secrets which have a certain relation. This feature will be explicitly used in the next section as well as in the protocols for secure multiparty computations.

4 Proving Polynomial Relations among Shared Secrets

A linear VSS can be seen as a linear commitment to the shared secret. Using techniques from [21] it is possible to very efficiently prove polynomial relations among shared secrets.

We denote by $[a]_i$ the pair of the verification function and the share the player P_i holds from the secret a.

For proving a linear relation among commitments one turns this relation into a set of linear functions which all must equal zero when evaluated on the committed values if and only if the relation holds. Proving that a given linear function evaluates to zero on committed values can be done by means of Proposition 1: using the linearity of the VSS scheme one computes from the given commitments a new commitment which represents the linear function evaluated on the given commitments and this new commitment is then opened to be zero.

To be able to prove arbitrary polynomial relations on committed values we will first restate a protocol from [21] which allows to compute a new commitment which represents the product of two given commitments. This protocol can directly be applied to the linear secret sharing scheme presented here. The protocol is called a *distributed one time multiplication proof (DOTMP)* and consists of two phases: a pre-distribution phase where the trusted center shares additional values among the players and a non interactive proof where the additional shared information is used to compute shares to the product of two shared values without reconstructiong these.

Protocol DOTMP

- Initialization: The trusted center verifiably shares (with the players) three random numbers l, l' and l'', such that $l'' = ll'$. Thus, each player P_i receives $[l]_i, [l']_i$ and $[l'' = ll']_i$
- Multiplication: Each player P_i player now holds shares to three random values l, l' and l'', such that $l'' = ll'$ as well as two shares $[a]_i$ and $[b]_i$ to the values a, b which are to be multiplied, to obtain a share $[ab]_i$ to ab each player P_i computes $[y]_i = [a]_i - [l]_i = [a - l]_i$ and $[y']_i = [b]_i - [l']_i = [b - l']_i$ and together with the other players reconstruct y and y'. Now P_i calculates $[ll']_i + y[l']_i + y'[l]_i + yy'$.

Using this protocol arbitrary polynomial relations can be proven analogously to the linear relations. The polynomial relations are turned into a set of multivariate polynomials which should simultaneously vanish on the committed values. To prove that a polynomial vanishes on given committed values a new commitment representing the polynomial evaluated at the given commitments is computed step by step using the above multiplication protocal as well as addition and scalar multiplication which are granted by the linearity of the VSS scheme. The new commitment is then opened to be zero. Summarizing the above we obtain:

Proposition 2. *For given shared values a, b, c and a constant $\lambda \in GF(q)$ it is possible to calculate shares for the value $ab + \lambda c$ if and only if $t + 1$ players cooperate.*

Especially it is possible to give zero knowledge proofs for arbitrary polynomial relations on shared values.

Proof. The security and correctness are obvious for the linear part and have only to be proven for multiplicative relations, i. e., for the DOTMP protocol:

We now analyze the security and correctness of our protocol for proving multiplicative relations among shares. To show that it is secure, note that the players only learns the values y and y', which give no information on a and b, since l and l' are random numbers.

To show the correctness of our protocol, we note that $[ll'] + y[l'] + y'[l] + yy' = [ll'] + (a-l)[l'] + (b-l')[l] + (a-l)(b-l') = [ll' + al' - ll' + bl - l'l + ab - lb - l'a + ll'] = [ab]$ (due to the linearity of the VSS).

By applying addition of shares and multiplication of shares we can obtain shares for arbitrary polynomial relations among shares. E.g. to prove that $mm' = m''$ the dealer lets the players compute $[mm'] - [m''] = [mm' - m'']$ and shows this to be zero.

For adding two shared values or for multiplication with a constant no-one has to know the shared values which are linearly transformed. Note that in DOTMP, too, no-one has to know the shared values which have to be multiplied in advance as the values y and y' are reconstructed in the protocol. Hence an arbitrary polynomial evaluation on shared values works iff $t+1$ players cooperate. This is optimal as $n-t$ players could abort the VSS scheme anyway. These linear transformations and multiplications on shared values will be the building blocks for the multiparty protocols presented in Section 6. To obtain *fair* protocols we need one more pre-distributed primitive: pre-distributed fair exchange of secrets.

5 Pre-distributed Fair Exchange of Secrets

In this section, we show how to use our VSS protocol so that n parties $P_1, \dots P_n$, each one holding a secret x_i, can exchange these secrets so that no cheating party can, at the end of the protocol, have substantially more advantage over an honest party. For definitions and more details see [9][13].

The basic idea here is that a trusted center pre-distributes, during a setup phase, verifiable secret sharing (commitments) which can be gradually disclosed. In the following protocols, a cheating party which leaves the protocol before it is terminated achieves an advantage of at most a polynomial fraction of a bit.

Each party P_i holds a secret x_i and each party has shared its secret with all the other players. So, besides his own secret, each player holds shares (including the verification information) of all the other players secrets. Denote the share of the secret x_i in possession of player P_j by $[x_i]_j$. A multiparty fair exchange of secrets in the commodity based model is a protocol where a trusted center pre-distributes some information to a set of players $P_1, \dots, P_n$ during a setup phase and later on, $P_1, \dots, P_n$ run a sub-protocol Fair Exchange. At the end of Fair Exchange the following conditions hold:

– If all the parties are honest and follow the protocol, each party will know all the $x_i, 1 \leq i \leq n$.
– A cheating party which leaves the protocol before it is terminated has an extra knowledge over honest parties of at a fraction of a bit which is polynomial in the security parameter k.

The protocol consists of two phases: Setup and Fair Exchange. We assume that the players have already verifiably shared their secrets using the protocol we described above. All computations are done over $GF(q)$.

Setup

– The trusted center sends to each player a random number r_i.
– The trusted center verifiable shares each r_i with all the players.
– Let $r_{i,1}, \dots, r_{i,p(k)}$ be a string of bits chosen at random such that each prefix $r_{i,1}, \dots, r_{i,l}$ of this string contains $l/p(k)$ bits of information on the random number $r_i \in GF(q)$. For $1 \leq k \leq p(k)$ and $1 \leq i \leq n$ the trusted center

verifiably shares $r_{i,k}$ with all the players. So, at the end of the setup phase, each player P_i possesses his own secret x_i; shares of all the other players secrets, $[x_j]_i, 1 \leq j \leq n$; a random value r_i and shares for each bit of the above binary string representation of each random number associated to the other players $[r_{i,k}], 1 \leq k \leq \log_2 p$ and $1 \leq i \leq n$. We call these data which is distributed during the setup phase a *one-time distributed fair exchange* (OTDFE).

Fair Exchange

- All the players compute shares to $a_i = x_i - r_i, 1 \leq i \leq n$. Due to the linearity of the VSS, each party P_j, $1 \leq j \leq n$ only computes $[x_i]_j - [r_i]_j = [x_i - r_i]_j = [a_i]_j$.
- All the players reconstruct a_i, $1 \leq i \leq n$. Thus, from now on, all the a_i are public values.
- For $k = 1$ to $p(k)$ do:
  - For $i = 1$ to n do:
    - Reconstruct the secret $r_{i,k}$. If $n - t$ or more parties refuse to reconstruct a r_{i_k} the protocol is aborted.
- If all the $r_{i,k}$ were successfully recovered, each party computes $r_i, 1 \leq i \leq n$. Together with the a_i, $1 \leq i \leq n$ it gives full knowledge of each secret x_i to all the parties.

From the construction it is obvious that we obtain the following result for the above fair exchange protocol:

Proposition 3. *Let $\{P_1, \ldots, P_n\}$ be a set of players holding correct shares of a secret s, then the above fair exchange protocol either terminates with output s for all honest players or any t-subset of players aborting the protocol can only have an advantage of a fraction of a bit which vanishes polynomially in the security parameter k.*

There is a more efficient variant of the above scheme. The bits $r_{i,1}, \ldots, r_{i,m}$ can be shared as elements of a **smaller** field than $GF(q)$. This increases the probability with which a party can successfully present faulty shares, but this does not compromise the asymptotic security. The advantage an aborting party may have over other parties is polinomial in k, whereas the probability of being able to successfully present faulty shares decreases exponentially in the length of the representation of $GF(q)$. Hence there is for a fixed maximal advantage for aborting parties a trade-off between the length of the expansion $r_{i,1} \ldots, r_{i,p(k)}$ chosen for the fair exchange and the size of the field over which the bits are shared. For this modification it is important that correctly guessing one value of the (smaller) field does not allow a party to present many faulty shares. To avoid this one can choose the verification information v_i of a player P_i newly for each shared bit $r_{i,j}$. Of course this implies that the shared bits $r_{i,1} \ldots, r_{i,p(k)}$ do not exhibit any linearity any more, but this is not needed for fair exchange anyway.

6 Secure Multiparty Computations

In this section we will sketch a simple way to obtain multiparty protocols for secure function evaluation from the VSS scheme presented. For details concerning multiparty protocols we refer to the literature, e. g. [1,14,15,13,19].

Intuitively a multiparty computation is a protocol by which n interacting Turing machines can map $n-$tuples of inputs (one input held by each party) into $n-$tuples of outputs (one held by each party). Such a computation will be considered secure if it is private, correct and fair [14], informally these properties are:

Private: No party learns anything more than what can be computed from the own input and the output of the protocol.

Correct: The output received by each party is guaranteed to be the output of the specified function.

Fair: Corrupted parties should receive an output iff honest parties do.

The fairness requirement is usually relaxed in the faulty majority scenario. We assume that the additional information a corrupted party has about the computation's output can be made arbitrarily small in a security parameter k.

The secure multiparty protocols presented here have four stages. A *setup phase* where the trusted center pre-distributes data. An *input phase* where the players receive inputs and commit to these by VSS. A *computation phase* where linear transformations and multiplications are performed on shared values, but no information about the inputs is revealed. And the *opening stage* during which the relaxed notion of fairness described above has to be ensured.

Setup Phase. In this stage, all the players contact the trusted center and receive pre-distributed verifiable secret sharing, DOTMP and OTDFEs. An upper bound on the number of commodities needed must be known in this stage. We call the union of these primitives a *One-Time Distributed Computation.*

Input StagePhase. The players receive inputs from $GF(q)$ and share their inputs with the given commodities for VSS as described in the main part of this paper. As each dealer is a participant of the secure computation as well this party has to compute a share of his own from the pre-distributed data.

Computation Phase. During the computation stage, the players evaluate an arithmetic circuit gate by gate using the linearity of the VSS for linear transformations and the DOTMP protocol for multiplications. Note that computations are necessary on intermediate results as well. Intermediate results are shared among the players, but not known to any player hence it is important that the linearity of the VSS and the DOTMP protocol can be used even if no-one knows the contents of the shared secrets involved.

Opening Phase. All players reconstruct the result of the computation. To ensure fairness the pre-distributed protocols for multiparty fair exchange can be used.

The security of the multiparty protocol can be derived from the security of each sub component of the protocol (VSS, DOTMP, OTDFEs).

References

1. D. Beaver and S. Goldwasser, Multiparty Computation with Faulty Majority, Proc. of FOCS, 1989, pp.468-473.
2. D. Beaver and S. Goldwasser, Multiparty Computation with Faulty Majority, Advances in Cryptology - CRYPTO 89, LNCS 435, 1989, pp.589-590.
3. D. Beaver: "Commodity-Based Cryptography (Extended Abstract)," STOC 1997: 446-455, 1997.
4. M. Ben-Or, S. Goldwasser and A. Wigderson, " Completeness Theorems for Non-Cryptographic Fault-Tolerant Distributed Computation," 20th STOC, pp. 1-10, 1988.
5. G. Blakely, "Safeguarding Cryptographic Keys," Proc. AFIPS, Vol. 48, pp. 313-317, NCC, June 1979.
6. C. Blundo, B. Masucci, D. R. Stinson, R. Wei, "Constructions and Bounds for Unconditionally Secure Non-interactive Commitment Schemes," manuscript, 2001.
7. B. Chor, S. Goldwasser, S. Micali and B. Awerbuch, " Verifiable Secret Sharing and Achieving Simultaneity in the Presence of Faults," 26th IEEE Symp. on Foundations of Computer Science, pages 383–395, 1985.
8. D. Chaum, C. Crepeau, and I. Damgard. Multiparty unconditionally secure protocols (extended abstract). In Proc. 20th ACM Symposium on the Theory of Computing (STOC), pp. 11-19, 1988.
9. R. Cleve: "Controlled Gradual Disclosure Schemes for Random Bits and Their Applications," CRYPTO 1989: 573-588, 1989.
10. C. Crepeau, J. van de Graaf, and A. Tapp, "Committed Oblivious Transfer and Private multiparty Computations," CRYPTO 1995: 110-123, 1995.
11. R. Cramer, I. Damgard, S. Dziembowski, M. Hirt, T. Rabin, "Efficient Multiparty Computations Secure Against an Adaptive Adversary," EUROCRYPT 99, pp. 311-326, 1999.
12. P. Feldman, "A practical scheme for non-interactive verifiable secret sharing," 28th IEEE Symp. on Foundations of Computer Science, 427–437, 1987.
13. S. Goldwasser, L. A. Levin: Fair Computation of General Functions in Presence of Immoral Majority. CRYPTO 1990: 77-93
14. O. Goldreich: Secure multiparty Computation, *lecture notes*, available from http://www.wisdom.weizmann.ac.il/~oded/pp.html
15. O. Goldreich, S. Micali, and A. Wigderson: "How to Play Any Mental Game or a Completeness Theorem for Protocols with Honest Majority," STOC 1987: 218-229, 1987.
16. G. Hanaoka G., J. Shikata, Y. Zheng, Imai H., "Unconditionally Secure Digital Signature Schemes Admitting Transferability," Proc. of Asiacrypt '2000, 130–142, 2000.
17. L. Lamport, R. Shostak, and M. Pease, "The Byzantine generals problem," ACM Trans. on Programming Languages and Systems, 4(3):382– 401, July 1982.

18. T. Matsumoto and H. Imai: “On the Key Pre-distribution Systems: A Practical Solution to the Key Distribution Problem,” CRYPTO 1987: 185-193, 1988.
19. S. Micali and P. Rogaway: “Secure Computation (Abstract),” CRYPTO 1991: 392-404, 1991.
20. R. J. McEliece and D. V. Sarwate, ”On sharing secrets and Reed-Solomon codes,” Communications of the ACM, 24:583–584, 1981.
21. A. Nascimento, J. Mueller-Quade, A. Otsuka, G. Hanaoka, H. Imai, ”Unconditionally Secure Homomorphic Pre-distributed Bit Commitment and Secure Two-Party Computations”, ISC’03
22. T. Pedersen, ”Non-interactive and information-theoretic secure verifiable secret sharing,” Advances in cryptology — CRYPTO ’91, volume 576 of Lecture Notes in Computer Science, pages 129–140. Springer-Verlag, 1991.
23. T. Rabin and M. Ben-Or, ”Verifiable secret sharing and multiparty protocols with honest majority,” Proc. ACM STOC ’89, pages 73–85, ACM Press, 1989.
24. R.L. Rivest, ”Unconditionally secure commitment and oblivious transfer schemes using concealing channels and a trusted initializer,” manuscript, 1999.
25. A. Shamir, ”How to Share a Secret,” Communications of the ACM, 22, pp. 612–613, 1979.
26. J. Shikata, G. Hanaoka, Y. Zheng, H. Imai, ”Security Notions for Unconditionally Secure Signature Schemes,” EUROCRYPT 2002, pages 434-449, 2002.
27. D. R. Stinson, R. Wei, ”Unconditionally Secure Proactive Secret Sharing Scheme with Combinatorial Structures,” Selected Areas in Cryptography, pages 200-214, 1999.
28. R. Safavi-Naini and H. Wang, “Multireceiver authentication codes: models, bounds, constructions and extensions,” Information and Computation, 151, pp.148-172, 1999.
29. M. Tompa and H. Wolf, ”How to share a secret with cheaters,” Journal of Cryptology, 1(2):133-138, 1988.