Basic Research in Computer Science

Efficient Multiparty Computations with Dishonest Minority

Ronald Cramer
Ivan B. Damgård
Stefan Dziembowski
Martin Hirt
Tal Rabin

BRICS Report Series **RS-98-36**

ISSN 0909-0878 **December 1998**

See back inner page for a list of recent BRICS Report Series publications. Copies may be obtained by contacting:

BRICS
Department of Computer Science
University of Aarhus
Ny Munkegade, building 540
DK–8000 Aarhus C
Denmark

Telephone: +45 8942 3360
Telefax: +45 8942 3255
Internet: BRICS@brics.dk

BRICS publications are in general accessible through the World Wide Web and anonymous FTP through these URLs:

`http://www.brics.dk`
`ftp://ftp.brics.dk`
This document in subdirectory `RS/98/36/`

Efficient Multiparty Computations with Dishonest Minority

Ronald Cramer* Ivan Damgård† Stefan Dziembowski‡
Martin Hirt§ Tal Rabin¶

Abstract

We consider verifiable secret sharing (VSS) and multiparty computation (MPC) in the secure channels model, where a broadcast channel is given and a non-zero error probability is allowed. In this model Rabin and Ben-Or proposed VSS and MPC protocols, secure against an adversary that can corrupt any minority of the players. In this paper, we first observe that a subprotocol of theirs, known as weak secret sharing (WSS), is not secure against an adaptive adversary, contrary to what was believed earlier. We then propose new and adaptively secure protocols for WSS, VSS and MPC that are substantially more efficient than the original ones. Our protocols generalize easily to provide security against general Q^2 adversaries.

1 Introduction

1.1 MPC and VSS

Consider a set of players each holding a private input, who wish to compute some agreed upon function of their inputs in a manner which would preserve

*ETH, Zürich
†Aarhus University, BRICS
‡Aarhus University, BRICS
§ETH, Zürich
¶T.J.Watson Research Center

the secrecy of their inputs. They need to carry out the computation even if some of the players may become corrupted and actively try to interfere with the computation.

As the first general solution to this problem, [GMW87] presented a protocol that allows n players to securely compute an arbitrary function even if a central adversary actively corrupts any $t < n/2$ of the players and makes them misbehave maliciously. However, this protocol assumes that the adversary is computationally bounded. In a model with secure and authenticated channels between each pair of players (the secure-channels model), [BGW88, CCD88] proved that unconditional security is possible if at most $t < n/3$ of the players are actively corrupted. This bound was improved in [RB89, Bea91] to $t < n/2$ by assuming the existence of a broadcast channel.

One important subprotocol of multiparty computation is Verifiable Secret Sharing (VSS) [CGMA85]. A VSS scheme for n players that is resilient against an adversary that actively corrupts t malicious players allows the *dealer* to share a secret among the players in such a way that the adversary obtains no information about the secret, and that the secret can be efficiently reconstructed, even if the corrupted players try to disturb the protocol.

1.2 Contributions

Rabin and Ben-Or [RB89, Rab94] proposed VSS and MPC protocols, secure against an adversary that can actively corrupt any minority of the players. In this paper, we observe that a subprotocol of theirs, known as weak secret sharing (WSS, a type of unconditionally secure commitment scheme), is not secure against an adaptive adversary, contrary to what was believed earlier (and claimed in [RB89, Rab94]). However, the VSS protocol of [RB89, Rab94] is in fact adaptively secure.

We propose new and adaptively secure protocols for WSS, VSS and MPC that are substantially more efficient than the original ones of [RB89] and later protocols by Beaver [Bea91]. To obtain error probability $2^{-k+O(\log n)}$ with n players, the VSS protocols of [RB89, Bea91] communicate $\Omega(k^3n^4)$ bits. Our VSS protocol is constant round and uses communication $O(kn^3)$ bits, to achieve the same error probability $2^{-k+O(\log n)}$.

This improvement is based in part on a more efficient implementation of *Information Checking Protocol*, a concept introduced in [RB89] which can be described very loosely speaking as a kind of unconditionally secure signature

scheme. Our implementation is linear meaning that for two values that can be verified by the scheme, and linear combination of them can also be verified with no additional information. When using our VSS in MPC, this means that linear computations can be done non-interactively, contrary to what the implementation of [RB89] (this property was also obtained in [Bea91], but with a less efficient Information Checking implementation).

An essential tool in MPC (provided in both [RB89] and [Bea91]) is a protocol that allows a player who has committed to values a, b, c using WSS, to show that $ab = c$ without revealing extra information. We provide a protocol for this purpose giving error probability 2^{-k} which is extremely simple.

Using methods recently developed in [CDM98], our protocols generalize easily to provide security against general Q^2 adversaries [HM97].

1.3 Outline

We first show that the Weak Secret Sharing (WSS) scheme of [RB89, Rab94] is not adaptively secure (Section 3). In Section 4, we propose an efficient implementation of Information Checking, and in Section 5, a scheme for Verifiable Secret Sharing (VSS) id developed. Based on these protocols, in Section 6 an efficient protocol for multiparty computation (MPC) is presented. Finally, in Section 7 an efficient protocol secure against general (non-threshold) adversaries is sketched.

2 Model and Definitions

In this paper, we consider the *secure-channels model with broadcast*, i.e. there are n players $P_1, \ldots, P_n$ who are pairwise connected with perfectly private and authenticated channels, and there is a broadcast channel. There is a *central adversary* with unbounded computing power who actively corrupts up to t players where $t < n/2$. To actively corrupt a player means to take full control over that player, i.e. to make the player (mis)behave in an arbitrary manner. The adversary is assumed to be *adaptive* (or dynamic), this means that he is allowed to corrupt players during the protocol execution (and his choice may depend on data seen so far), in contrast to a static adversary who only corrupts players before the protocol starts. The security of the

presented protocols is unconditional with some *negligible error probability*, which is expressed in term of a *security parameter* k. The protocols operate in a finite field $K = GF(q)$, where $q > \max(n, 2^k)$.

2.1 Definition of Information Checking

A scheme for *information checking* (IC) consists of three protocols.

Distr(D, INT, R, s) is initiated by the dealer D. In this phase D hands the secret s to the intermediary INT and some auxiliary data to both INT and the recipient R.

AuthVal(INT, R, s) is initiated by INT and carried out by INT and R. In this phase INT ensures that in the protocol RevealVal R (if honest) will accept s, the secret held by INT.

RevealVal(INT, R, s) is initiated by INT and carried out by INT and R. In this phase R receives a value s' from INT, along with some auxiliary data, and either accepts s' or rejects it.

The IC scheme has the following properties:

Correctness:

A. If D, INT, and R are uncorrupted, and D has a secret s then R will accept s in phase RevealVal.

B. If INT and R are honest then after the phases Distr and AuthVal the INT player knows a value s such that R will accept s in the phase RevealVal (except with probability 2^{-k}).

C. If D and R are uncorrupted, then in phase RevealVal with probability at least $1 - 2^{-k}$, player R will reject every value s' different from s.

Secrecy:

D. The information that D hands R in phase Distr is distributed independently of the secret s. (Consequently, if D and INT are uncorrupted, and INT has not executed the protocol RevealVal, R has no information about the secret s.)

Definition 1 *An IC scheme is a triple* (Distr, AuthVal, RevealVal) *of protocols that satisfy the above properties A. to D.*

2.2 Definition of WSS

An intuitive explanation for a *weak secret-sharing (WSS) scheme* is that it is the non-computational analog of a computational commitment. It exhibits the same properties, i.e. it binds the committer to a single value after the sharing phase Sh, yet the committer can choose not to expose this value in the reconstruction phase Rec. A WSS scheme for sharing a secret $s \in K$ consists of the two protocols Sh and Rec that satisfy the following properties, with an allowed error probability 2^{-k}:

- *Termination*: If the dealer D is honest then all honest player will complete Sh, and if the honest players invoke Rec, then each honest player will complete Rec.
- *Secrecy*: If the dealer is honest and no honest player has yet started Rec, then the adversary has no information about the shared secret s.
- *Correctness*: Once all currently uncorrupted players complete protocol Sh, there exists a *fixed* value, $r \in K \cup \{\mathsf{NULL}\}$, such that the following requirements hold:
  1. If the dealer is uncorrupted throughout protocols Sh and Rec then r is the shared secret, i.e. $r = s$, and each uncorrupted player will outputs r at the end of protocol Rec.
  2. If the dealer is corrupted then each uncorrupted player outputs either r or NULL upon completing protocol Rec.

Definition 2 *A t-secure WSS scheme for sharing a secret $s \in K$ is a pair* $(\mathsf{Sh}, \mathsf{Rec})$ *of two protocols that satisfy the above properties even in the presence of an active adversary who corrupts up to t players.*

2.3 Definition of VSS

A pair $(\mathsf{Sh}, \mathsf{Rec})$ of protocols is a *verifiable secret-sharing (VSS) scheme* if it satisfies a stronger correctness property, with an allowed error probability 2^{-k}.:

- *Correctness*: Once all currently uncorrupted players complete protocol Sh, there exists a *fixed* value, $r \in K$, such that the following requirements hold:

1. If the dealer is uncorrupted throughout protocol Sh then r is the shared secret, i.e. $r = s$, and each uncorrupted player outputs r at the end protocol Rec.
2. If the dealer is corrupted then each uncorrupted player outputs r upon completing protocol Rec.

Definition 3 *A t-secure VSS scheme for sharing a secret $s \in K$ is a pair* $(\mathsf{Sh}, \mathsf{Rec})$ *of two protocols that satisfy the Termination and the Secrecy property of WSS, and the above, stronger, Correctness property, even in the presence of an active adversary who corrupts up to t players.*

2.4 Definition of MPC

The goal of multiparty computation (MPC) is to evaluate an agreed function $g : K^n \to K$, where each player provides one input and receives the output. The privacy of the inputs and the correctness of the output is guaranteed even if the adversary corrupts any t players. For a formal definition for security see [GL90, MR91, Bea91, Can98, MR98].

3 Adaptive Security of WSS in [RB89]

In this section we describe an adaptive attack against the weak secret-sharing scheme (WSS) of Rabin and Ben-Or [RB89, Rab94]. The attack can be derived due to the ability of the adaptive adversary to choose the set of corrupted players at any point in the protocol. The attack will only work when $t > n/3$. It is important to note that this attack applies only to the WSS protocol of [RB89] as a stand-alone protocol, and does not apply to their VSS scheme, although it uses the WSS as a subprotocol.

In order to explain the attack we present a simplified protocol of the [RB89] protocol which assumes digital signatures. It is in essence the same protocol but with many complicating (non relevant) details omitted.

WSS Share (Sh)

The dealer chooses a random polynomial $f(x)$ of degree t, such that $f(0) = s$ the secret to be shared, and sends the share $s_i = f(i)$ with his signature for s_i to each player P_i.

WSS Reconstruct (Rec)

1. Every player reveals his share s_i and the signature on s_i.
2. If *all* properly signed shares $s_{i1}, ...s_{ik}$ for $k \geq t$ interpolate a single polynomial $f'(x)$ of degree at most t, then the secret is taken to be $f'(0)$ otherwise no secret is reconstructed.

The definition of WSS requires that at the end of Sh a single value $r \in K \cup \{\mathsf{NULL}\}$ is set so that only that value (or NULL) will be reconstructed in Rec. We will show that under an adaptive adversary this requirement does not hold in the above described protocol.

The attack proceeds as follows: in the protocol Sh the adaptive adversary corrupts the dealer causing him to deviate from the protocol. The dealer chooses two polynomials $f_1(x)$ and $f_2(x)$ both of degree at most t, where $f_1(0) \neq f_2(0)$, and $f_1(i) = f_2(i)$ for $i = 1, \ldots, t$. The players $i = 1, \ldots, t$ receive $f_1(i) = f_2(i)$ as their share, the players $i = t+1, \ldots, 2t$ receive $f_1(i)$ as their share, and the remaining (at most t) players receive $f_2(i)$ as their share. All shares are given out with valid signatures.

In Rec the adversary can decide whether to corrupt $P_{t+1}, ..., P_{2t}$ thus forcing the secret to be $f_2(0)$ or to corrupt $P_{2t+1}, ..., P_n$ and thus force the secret to be $f_1(0)$. Hence it is clear that at the end of Sh there is not a *single* value which can be reconstructed in Rec. The decision on which value to reconstruct can be deferred by the adversary until the reconstruction protocol Rec is started.

Therefore the basic problem with stand-alone WSS is that it is not ensured that all honest players are on the same polynomial immediately after distribution. But when using it inside the VSS of [RB89], this property is ensured as a side effect of the VSS distribute protocol, hence the VSS protocol works correctly.

4 The Information Checking Protocol

In this section we present protocols that satisfy Definition 1 for information checking (cf. Section 2.1). They provide the same functionality as the check vector protocol from [RB89, Rab94] and the Time Capsule protocol from [Bea91]. However, our implementation of Information Checking also possesses an additional property which will be utilized later in the paper. The

property is the following: given two secrets and the ability to verify them, it is possible to also verify any linear combination of the two, without any additional information.

The basic idea for the construction will be that the secret and the verification information will all lie on a polynomial of degree 2, where the secret will be the value at the origin. INT will receive two points on this polynomial (one being the secret) and R will receive two additional points. Thus, INT will be able to authenticate the secret to R only if it will be able to give R two points which complete the values held by R into a polynomial of degree 2. We will see that such a construction enables to verify linearly dependent secrets. We describe our protocols in the following.

Definition 4 *A vector* $\vec{e} = (e_0, ..., e_{n-1}) \in K^n$ *is* t-consistent *if there exists a polynomial* $w(x)$ *of degree at most* t *such that* $w(i) = e_i$ *for* $0 \leq i < n$.

Protocol Distr(D, INT, R, s)**:**

The dealer D chooses two 2-consistent vectors of length 4. The first one is constructed by setting $e_0 = s$, choosing two random values $e_2, e_3 \in K$ and computing the last value e_1 to satisfy the requirement of 2-consistency. The second vector $\vec{r}$ is chosen at random. D hands the pairs (e_0, e_1) and (r_0, r_1) to the intermediary INT and the pairs (e_2, e_3) and (r_2, r_3) to the recipient R.

It is easy to see that the protocol Distr (together with RevealVal below) ensures Properties A, C and D as required in the definition — but not B (i.e., all properties are satisfied if the dealer is honest). The next protocol is designed to ensure Property B as well, without affecting A, C and D.

Protocol AuthVal(INT, R, s)**:**

1. INT chooses a random element $c \in K$ and broadcasts it.
2. R broadcasts $(a_2, a_3) = c \cdot (e_2, e_3) + (r_2, r_3)$. Now the dealer decides whether these values agree with the data sent to R before. If they do then he broadcast approval and the protocol goes to 3. Otherwise the dealer broadcasts that R should always accept $s(= e_0)$.
3. INT checks whether $(c \cdot e_0 + r_0, c \cdot e_1 + r_1, a_2, a_3)$ is 2-consistent. If yes, then the protocol is finished, otherwise R complaints and D has to broadcast the value of $s(= e_0)$. Now s is publicly known, and the RevealVal phase will be trivial.

Protocol RevealVal(INT, R, s)**:**

1. INT sends to R the values (e_0, e_1).
2. R checks whether (e_0, e_1, e_2, e_3) is 2-consistent and if they are R accepts the value e_0 as the value handed out by D.

Lemma 1 *The protocols* (Distr, AuthVal, RevealVal) *described above satisfy Definition 1 for Information Checking (Section 2.1).*

Proof We show that each property is satisfied:

A. Can be easily verified from the protocol.

B. R will reject e_0 only if $\vec{e} = (e_0, e_1, e_2, e_3)$ is not 2-consistent. Assume that this is the case, which means that $\vec{e}$ is 3-consistent. It is easy to see that for a fixed $\vec{r}$ there exists at most one value of c such that $c \cdot \vec{e} + \vec{r}$ is 2-consistent, hence the cheating dealer has a probability of $1/|K|$ of having $\vec{e}$ be 3-consistent, and the test carried out by INT be successful.

C. If INT chooses a value $s' \neq s = e_0$ then he must find e_1' such that (s', e_1', e_2, e_3) is 2-consistent. For a given (e_2, e_3) only one value e_1' will be 2-consistent. INT has no knowledge of (e_2, e_3), and all pairs are equally likely. Thus, the probability that R will accept s' is $1/|K|$.

D. The information that R holds is independent of the secret.

□

In the sequel we will use the Information Checking Protocol as "digital signatures" in the following way. Protocol Distr will be carried out by the dealer D with intermediary INT and the receiver being each player $P_1, ..., P_n$, each with respect to the same value s. Next, the AuthVal protocol will be performed by INT and each player P_i. Then, in protocol RevealVal INT will broadcast s and the authentication information, and if $t + 1$ players accept the value s then we shall say that the "signature" has been confirmed. We shall call these signatures IC-signatures. It should be noted that these do not capture all the properties of digital signatures, and more specifically, if D gives INT a "signature" only INT can verify it to the other players. Thus, we use these IC-signatures as a specific signature from D to INT, and we denote an IC-signature from D to INT on a value a as $\sigma_a(D, INT)$.

5 Verifiable Secret Sharing

We now present our simplified VSS protocol. For ease of presentation we shall describe the protocol first using digital signatures and then proceed to show that the properties which we are utilizing of the digital signatures can be achieved using Information Checking signatures (IC-signatures). The protocol is based on the bivariate solution of Feldman [FM88, BGW88] (omitting the need for error correcting codes). The protocol will use our new variant of Information Checking which will provide us with high efficiency.

The intuition behind the construction is that the secret will be shared using an $n \times n$ matrix of values, where each row and column is t-consistent. The dealer will commit himself to all these values by signing each value in the matrix. Thus, if he did not act properly this fact would be exposed using the signatures. The consistency property will be verified by the players together. Hence we are guaranteed that all the values held by (yet) uncorrupted players are consistent and define a single secret[1]. In order to prevent the adversary from corrupting the secret at reconstruction time, we also require that each player sign all the values which he holds in a given row. And thus no new values can be injected into the computation in the reconstruction.

The signature given from A to B for the value s will be denoted by $\sigma_s(A, B)$. (We make the signature dependent on B since in the real protocol the IC Protocol is used.)

VSS Share (Sh)

1. The dealer D chooses a random bivariate polynomial $f(x, y)$ of degree at most t in each variable, such that $f(0, 0) = s$. Let $s_{ij} = f(i, j)$. The dealer sends to player P_i the values $a_{1i} = s_{1i}, \ldots, a_{ni} = s_{ni}$ and $b_{i1} = s_{i1}, \ldots, b_{in} = s_{in}$. For each value a_{ji}, b_{ij} D attaches a digital signature $\sigma_{a_{ji}}(D, P_i)$, $\sigma_{b_{ij}}(D, P_i)$.

2. Player P_i checks that the two sets $a_{1i}, \ldots, a_{ni}$ and $b_{i1}, \ldots, b_{in}$ are t-consistent. If the values are not t-consistent, P_i broadcasts these values with D's signature on them. If a player hears a broadcast of inconsistent values with the dealer's signature then D is disqualified and execution is halted.

[1] So far, this results in a WSS which is secure against an adaptive adversary.

3. P_i sends a_{ji} and a signature which he generates on a_{ij}, $\sigma_{a_{ji}}(P_i, P_j)$ privately to P_j.

4. Player P_i compares the value a_{ij} which he received from P_j in the previous step to the values b_{ij} received from D. If there is an inconsistency or if he did not receive a value then P_i broadcasts $b_{ij}, \sigma_{b_{ij}}(D, P_i)$.

5. Player P_i checks if P_j broadcasted a value $b_{ji}, \sigma_{b_{ji}}(D, P_j)$ which is different than the value a_{ji} which he holds. If such a broadcast exists then P_i broadcasts $a_{ji}, \sigma_{a_{ji}}(D, P_i)$.

6. If for an index pair (i, j) a player hears two broadcasts with signatures from the dealer on different values, then D is disqualified and execution is halted.

VSS Reconstruct (Rec)

1. Player P_i broadcasts the values $b_{i1}, ..., b_{in}$ with the signature for value b_{ij} which he received from player P_j. (If he did not receive a signature from P_j in the protocol Sh then he had already broadcasted that value with a signature from D.)

2. Player P_i checks whether player P_j's shares broadcasted in the previous step are t-consistent and all the signatures are valid. If not then P_j is disqualified.

3. The values of all non-disqualified player are taken and interpolated to compute the secret.

Before we proceed to state and prove our theorem, we note that the "real" version of our protocols with information theoretic security is obtained by replacing the digital signatures with IC-signatures, i.e. the creation of a signatures by executions of Distr and AuthVal protocols, and broadcasting of signed messages by executions of RevealVal protocols. The basic reason why this works is that the only properties of signatures we rely on are captured by the properties A.-D. of Information Checking protocols.

Theorem 1 *The above protocols* (Sh, Rec) *satisfy Definition 3 for VSS protocols.*

Proof We proof that each required property is satisfied:

Secrecy. Observe that in Steps 2–6, the adversary learns nothing that he was not already told in Step 1. Thus the claim follows immediately from the properties of a bi-variate polynomial of degree t and the properties of the Information Checking.

Termination. From examining the protocol it is clear that the dealer D can be disqualified only if data which he shared is inconsistent, assuming that the players cannot forged the dealers signatures. Thus, an honest dealer will be disqualified at most with probability $O(2^{-k})$.

Correctness First we will show that there is a fixed value r. Define r to be the secret which interpolates through the shares held by the set of the first $t+1$ players who have not been corrupted during Sh. Their shares are trivially t-consistent, and with probability at least $1-O(2^{-k})$, there are correct signatures for their shares, and thus the value r is well defined with an underlying implicit polynomial $f'(x, y)$. Let us look at another uncorrupted player outside this set. He has corroborated his shares with all these $t+1$ players and has not found an inconsistency with them. Thus, his shares interpolate (at the minimum) through $f'(x, y)$ and hence are at least t-consistent. But this player has also verified that his shares are t-consistent. Hence, when this player's shares are added to the initial set of players' shares the set remains t-consistent, thus defining the same secret r. Now we examine the two correctness conditions:

1. It is easy to see that if D is uncorrupted then this value $r = s$.
2. A value different than r will be interpolated (or the reconstruction will fail) only if a corrupted player would be able to introduce values which are inconsistent with the values held by the honest players. A corrupted player succeeded doing it only when he was not disqualified in Step 2. of the reconstruction procedure. This means that he was able to produce a set of n values which are t-consistent, and for each value to have a signature from the appropriate player to which it relates. Clearly, $t+1$ of these signatures must be from still uncorrupted players. We have already shown that these players' shares lie on $f'(x, y)$, thus if the corrupted

player's shares are t-consistent they must define lie on $f'(x, y)$ as well. Therefore the adversary cannot influence the value of the revealed secret.

□

Efficiency By inspection of the VSS distribution protocol Sh, one finds that n^2 field elements are distributed from D, and each of these are authenticated using Distr and AuthVal a constant number of times. Executing Distr and AuthVal requires communicating a constant number of field elements for each player, and so we find that the total communication is $O(n^3 k)$ bits, for an error probability of $2^{-k+O(\log n)}$.

6 Multiparty Computation

Based on the VSS scheme of the previous section, we now build a multiparty computation protocol. Based on the [BGW88] paradigm it is known that it is sufficient to devise methods for adding and multiplying two shared numbers.

Note that in our case (contrary to e.g. [BGW88]) a VSS of a value a consists not only of the shares $a_1, \ldots, a_n$ where a_i is held (in fact implicitly) by P_i, it is explicitly held by P_i via the subshares $a_{i1}, \ldots, a_{in}$ where a_{ij} is held also by player P_j, and P_i has a IC-signature from P_j on that value. This structure and the IC-signatures are required for the reconstruction. Thus, if we wish to compute the sum/multiplication of two secrets we need to have the resultant in this same form.

We will prove the following theorem in the next two subsections.

Theorem 2 *Assume the model with a complete network of private channels between n players and a broadcast channel. Let C be any arithmetic circuit over the field K, where $|K| > max(n, \log k)$ and k is a security parameter. Then there is a multiparty computation protocol for computing C, secure against any adaptive adversary corrupting less than $n/2$ of the players. The complexity of this protocol is $O(n^2|C|)$ VSS protocols with error probability $2^{-k+O(\log n)}$, where $|C|$ is the number of gates in C. This amounts to $O(|C|kn^5)$ bits of communication.*

6.1 Addition

Addition is straightforward: For two secrets a and b shared with (implicit) shares $a_1, ..., a_n$ and $b_1, ..., b_n$, all the subshares, and their appropriate IC-signatures, each player P_i needs to add his two (implicit) shares a_i and b_i which means that he needs to hold a IC-signature from P_j for $a_{ij} + b_{ij}$. But this is immediately achieved as the linear combination of two IC-signatures results in an IC-signature for the linear combination of the values signed. Thus, we have computed the addition of two shared secrets.

6.2 Multiplication

Multiplication is slightly more involved. Assume that we have two secrets a and b with (implicit) shares $a_1, ..., a_n$ and $b_1, ..., b_n$ and all the subshares and their appropriate IC-signatures. We apply the method from [GRR98]. This method calls for every player to multiply his shares of a, resp. b and to share the result of this using VSS. This results in n VSS's and a proper sharing of the result c can be computed as a fixed linear combination of these (i.e. each player computes a linear combination of his shares from the n VSS's). Since our VSS is linear, like the one used in [GRR98], the same method will work for us, provided we can show that player P_i can share a secret c_i using VSS, such that it will hold that $c_i = a_i b_i$ and to prove that he has done so properly. If P_i fails to complete this process the simplest solution for recovery is to go back to the start of the computation, reconstruct the inputs of P_i, and redo the computation, this time simulating P_i openly. This will allow the adversary to slow down the computation by at most a factor linear in n.

In order to eliminate subindices let us recap our goal stated from the point of view of a player D. He needs to share a secret c using VSS which satisfies that $c = ab$. The value a is shared via subshares $a_1, ..., a_n$ (lying on a polynomial f_a, say) where a_i is held by player P_i and D holds an IC-signature of this value from P_i. The same holds for the value b (with a polynomial f_b).

1. D shares the value $c = ab$ using the VSS Share protocol. Let f_c be the polynomial defined by this sharing.

2. D chooses a random $\beta \in K$ and he shares β and βb. The sharing of these values is very primitive. D chooses a polynomial $f_\beta(x) =$

$\beta_t x^t + ... + \beta_1 x + \beta$ and gives player P_i the value $f_\beta(i)$ and an IC-signature on this value. A player complains if he did not receive a share and a signature, and the dealer exposes these values. The same is done for βb (with a polynomial $f_{\beta b}$).

3. The players jointly generate, using standard techniques, a random value r, and expose it.

4. D broadcast the polynomial $f_1(x) = r f_a(x) + f_\beta(x)$.

5. Player P_i checks that the appropriate linear combination of his shares lies on this polynomial, if it does not he exposes his signed share $f_\beta(i)$ and requires the dealer to expose the IC-signature which the dealer holds generated by P_i for the value a_i. If the dealer fails then D is disqualified.

6. If the dealer has not been disqualified each player locally computes $r_1 = f_1(0)$.

7. D broadcasts the polynomial $f_2(x) = r_1 f_b(x) + f_{\beta b}(x) - r f_c(x)$.

8. Each player checks that the appropriate linear combination of his shares lies on this polynomial, if it does not he exposes his signed share $f_{\beta b}(i)$ and $f_c(i)$ and requires the dealer to expose the IC-signature which the dealer holds generated by P_i for the value b_i. If the dealer fails then D is disqualified.

9. If D has not been disqualified P_i verifies that $f_2(0) = 0$, and accepts the sharing of c, otherwise D is disqualified.

The security of the protocol is guaranteed by the following lemma.

Lemma 2 *Executing the above protocol for sharing $c = ab$ does not give the adversary any information that he did not know before.*

Proof Wlog we can assume that the dealer is honest. Thus all the values revealed during the protocol look random to the adversary (except of the polynomial f_2 which is a random polynomial such that $f_2(0) = 0$). Therefore the adversary learns nothing. □

Lemma 3 *If $c \neq ab$ in the above protocol, then for a given honest player P_{j_0} the probability that dealer succeeds to perform the above procedure for $j = j_0$ is at most $\frac{1}{|K|}$.*

Proof Suppose there exist two distinct challenges r_1 and r_1' such that if any of them is chosen in in Step 3. then D is not disqualified in the next rounds. Step 4. guarantees that honest players have consistent shares of f_β, since we open f_1 and we know f_a is consistent. So there is a well-defined value β shared by f_β. In the same way Step 7 guarantees that $f_{\beta b}$ is consistent, so it defines some value z (which may or may not be βb). Now from Step 4., $r_1 = ra + \beta$ and $r_1' = r'a + \beta$, so from Step 7., we get $(ra + \beta)b + z + rc = 0 = (r'a + \beta)b + z + r'c$ and we conclude that $ab = c$. □

7 General Adversaries

It is possible to go beyond adaptive security against any dishonest *minority*, by considering *general*, i.e. not necessarily threshold *adversaries* [HM97]. Our results in this paper extend to this general scenario in a number of ways, following ideas developed in [CDM98].

First, by replacing Shamir Secret Sharing by Monotone Span Program Secret Sharing [KW94] in our WSS, we immediately obtain WSS protocols secure against any Q^2-adversary [HM97], with communication and computation polynomial in the monotone span program complexity of the adversary [CDM98].

Building on the linearity of this WSS and Monotone Span Program Secret Sharing, we can construct efficient VSS (with negligible, but non-zero error) secure against any Q^2-adversary.

Yet another approach is suggested by a method from [CDM98], that first transforms a linear secret sharing scheme into one that is "locally verifiable", i.e. the honest players can check by pairwise verifications that they received consistent shares. If we apply this transformation to a linear secret sharing scheme with a "built-in" check-vector mechanism, we can construct the desired VSS in a way similar to the construction of efficient VSS given in [CDM98], which has zero error probability but offers security against any Q^3-adversary.

Regardless of which of these approaches we follow, the resulting VSS facilitates multi-party computation secure against any Q^2-adversary if we base the construction of VSS on Monotone Span Programs with Multiplication [CDM98]. As far as general adversaries are concerned, security against Q^2-adversaries is the maximum attainable level of security.

Both of the constructions give complexity $O(km^4)$ bits, where m is the size of the monotone span program.[2]

much more

References

[Bea91] D. Beaver: *Secure Multiparty Protocols and Zero-Knowledge Proof Systems Tolerating a Faulty Minority*, J. Cryptology, vol. 4 (1991) pp. 75-122.

[BGW88] M. Ben-Or, S. Goldwasser, A. Wigderson: *Completeness Theorems for Non-Cryptographic Fault-Tolerant Distributed Computation*, Proc. ACM STOC '88, pp. 1–10.

[Can98] R. Canetti: *Security and Composition of Multi-Party Cryptographic Protocols*, June 98, Manuscript.

[CCD88] D. Chaum, C. Crépeau, I. Damgård: *Multiparty Unconditionally Secure Protocols (Extended Abstract)*, Proc. ACM STOC '88, pp. 11–19.

[CGMA85] B. Chor, S. Goldwasser, S. Micali, and B. Awerbuch. Verifiable Secret Sharing and Achieving Simultaneity in the Presence of Faults. In *Proceeding 26th Annual Symposium on the Foundations of Computer Science*, pages 383–395. IEEE, 1985.

[CDM98] R. Cramer, I. Damgård and U. Maurer: *Span Programs and General Secure Multiparty Computations*, manuscript, 1998.

[2] [SS98] are less efficient ($O(k^3m^4)$ bits) by directly applying the ideas from [CDM98] to [Rab94], i.e. replacing Shamir's secret sharing by the Monotone Span Programs with Multiplication from [CDM98].

[FM88] P. Feldman, S. Micali: *Optimal algorithms for Byzantine agreement*, Proc. ACM STOC '88, pp. 148–161.

[GRR98] R. Gennaro, M. Rabin, T. Rabin, *Simplified VSS and Fast-Track Multiparty Computations with Applications to Threshold Cryptography*, Proc. ACM PODC '98.

[GMW87] O. Goldreich, S. Micali, A. Wigderson: *How to Play any Mental Game — A Completeness Theorem for Protocols with Honest Majority*, Proc. ACM STOC '87, pp. 218–229.

[GL90] S. Goldwasser, L. Levin: *Fair Computation of General Functions in Presence of Immoral Majority*, CRYPTO '90.

[HM97] M. Hirt, U. Maurer: *Complete Characterization of Adversaries Tolerable in General Multiparty Computations*, Proc. ACM PODC '97, pp. 25–34.

[KW94] M. Karchmer, A. Wigderson: *On Span Programs*, Proc. of Structure in Complexity, 1993.

[MR91] S. Micali, P. Rogaway: *Secure computation*, CRYPTO '91, pp. 392–404.

[MR98] S. Micali, P. Rogaway: *Secure Computation: The Information Theoretic Case*, June 98, Manuscript.

[Rab94] T. Rabin: *Robust Sharing of Secrets when the Dealer is Honest or Cheating*, J. ACM, 41(6):1089-1109, 1994.

[RB89] T. Rabin, M. Ben-Or: *Verifiable Secret Sharing and Multiparty Protocols with Honest majority*, Proc. ACM STOC '89, pp. 73–85.

[Sha79] A. Shamir: *How to Share a Secret*, Communications of the ACM 22 (1979) 612–613.

[SS98] A. Smith and A. Stiglic: *Multiparty Computations Unconditionally Secure against Q^2 Adversaries*, manuscript, 1998.

[Yao82] A.C. Yao: *Protocols for secure computations*, Proc. IEEE FOCS '82, pp. 160–164.

Recent BRICS Report Series Publications

RS-98-36 **Ronald Cramer, Ivan B. Damgård, Stefan Dziembowski, Martin Hirt, and Tal Rabin.** ***Efficient Multiparty Computations with Dishonest Minority*****. December 1998. 19 pp. To appear in** ***Advances in Cryptology: International Conference on the Theory and Application of Cryptographic Techniques*****, EUROCRYPT '99 Proceedings, LNCS, 1999.**

RS-98-35 **Olivier Danvy and Zhe Yang.** ***An Operational Investigation of the CPS Hierarchy*****. December 1998.**

RS-98-34 **Peter G. Binderup, Gudmund Skovbjerg Frandsen, Peter Bro Miltersen, and Sven Skyum.** ***The Complexity of Identifying Large Equivalence Classes*****. December 1998. 15 pp.**

RS-98-33 **Hans Hüttel, Josva Kleist, Uwe Nestmann, and Massimo Merro.** ***Migration = Cloning ; Aliasing (Preliminary Version)*****. December 1998. 40 pp. To appear in** ***6th International Workshop on the Foundations of Object-Oriented*****, FOOL6 Informal Proceedings, 1998.**

RS-98-32 **Jan Camenisch and Ivan B. Damgård.** ***Verifiable Encryption and Applications to Group Signatures and Signature Sharing*****. December 1998. 18 pp.**

RS-98-31 **Glynn Winskel.** ***A Linear Metalanguage for Concurrency*****. November 1998. 21 pp.**

RS-98-30 **Carsten Butz.** ***Finitely Presented Heyting Algebras*****. November 1998. 30 pp.**

RS-98-29 **Jan Camenisch and Markus Michels.** ***Proving in Zero-Knowledge that a Number is the Product of Two Safe Primes*****. November 1998. 19 pp, to appear in** ***Advances in Cryptology: International Conference on the Theory and Application of Cryptographic Techniques*****, EUROCRYPT '99 Proceedings, LNCS, 1999.**

RS-98-28 **Rasmus Pagh.** ***Low Redundancy in Dictionaries with*** $O(1)$ ***Worst Case Lookup Time*****. November 1998. 15 pp.**