Verifiable Secret Sharing and Multiparty Protocols with Honest Majority (Extended Abstract)

Tal Rabin Michael Ben-Or*
Institute of Mathematics and Computer Science
The Hebrew University, Jerusalem, Israel

Abstract

Under the assumption that each participant can broadcast a message to all other participants and that each pair of participants can communicate secretly, we present a verifiable secret sharing protocol, and show that any multiparty protocol, or game with incomplete information, can be achieved if a majority of the players are honest. The secrecy achieved is unconditional and does not rely on any assumption about computational intractability. Applications of these results to Byzantine Agreement are also presented.

Underlying our results is a new tool of Information Checking which provides authentication without cryptographic assumptions and may have wide applications elsewhere.

Introduction

Comparing the recent unconditional results on multiparty protocols of Ben-Or, Goldwasser and Wigderson [BGW] and of Chaum, Crepeau and Damgard [CCD], to the previous cryptographic results of Goldriech, Micali and Wigderson [GMW], based on the existence of secure Trapdoor functions, we find the following: On the one hand, the cryptographic results are stronger since they require only a majority of the n participating players to be honest, while the unconditional results require that more than $\frac{2}{3}n$ of the players be honest. On the other hand, the unconditional results are both simpler and stronger since their security is "Information Theoretic" and does not depend on any assumptions about computational intractability. Moreover, the results of [BGW] with up to $t \leq n/3$ faulty players guarantee absolute correctness of the results, and do not allow even an exponentially small probability of error. Such exponentially small errors are inherent to any cryptographic result based on Trapdoor functions.

The one third limit on the number of unreliable participants in the unconditional results mentioned above is necessary since otherwise even Byzantine agreement is impossible. It is natural to ask, however, what can be done if we ensure agreement by assuming the existence of a broadcast channel? Likewise, by [D], Byzantine agreement is still possible in incomplete networks with $2t+1$ connectivity, and we can ask what multiparty protocols can be achieved under these conditions.

To understand why a broadcast channel doesn't seem to help that much, we note a theorem of [BGW] stating that the absolutely errorless results *cannot* be extended to tolerate $t \geq n/3$ faulty players even with a broadcast channel. Moreover, as stated informally in [CCD], the broadcast channel does not allow an absolutely errorless solution even to the Verifiable Secret Sharing problem (VSS) with $t \geq n/3$ faulty players. We formally prove here that any solution must allow some error probability $\epsilon > 0$, though ϵ may be only exponentially small (deleted from this extended abstract). Since the VSS protocol is an essential component of any secure multiparty protocol these negative results led many researchers to conjecture that some cryptographic assumptions are indeed needed to handle t faulty players for $n/3 \leq t < n/2$, even with a broadcast channel.

In a similar way, the VSS protocols of [CCD] and [BGW] were used by Feldman and Micali [FM]

*Research supported by the Israel Academy of Sciences and by United States – Israel Binational Science Foundation Grant BSF-87-00082

and Feldman [F] to achieve constant expected time non cryptographic Byzantine agreement protocols in the synchronous and asynchronous cases respectively. Here again there are some intriguing gaps. In the synchronous case assuming a complete communication network, the protocols of [FM] can tolerate up to $t < n/3$ faulty players, but leave the general network case to cryptographic protocols. In the asynchronous case the non-cryptographic solution of [F] allows only $t < n/4$ faulty players, and relies on cryptographic VSS protocols to handle up to $t < n/3$ faults. Feldman has also raised the conjecture that $t < n/4$ is the limit for any unconditional asynchronous VSS protocol.

The main results of this paper close most of these gaps. We show

Main Theorem: *Assuming the existence of a broadcast channel and private communication channels between each pair of n players, there exists a (polytime) Verifiable Secret Sharing protocol tolerating up to $t < n/2$ faulty players. The secrecy achieved is unconditional, and the probability of error is exponentially small.*

A new and very practical idea of *Information Checking* is introduced in this paper. Information Checking is what enables our solution, and it may have wide applications elsewhere. It is a non-cryptographic weaker version of Digital Signatures. Weaker in the sense that not all securities provided by Digital Signatures are supplied by Information Checking, but it gives exactly the right amount of authentication power we need.

As a simple application of our information checking technique we show, (Compare Tompa and Woll [TW]),

Theorem 1: (VSS with Honest Dealer) *Let $n \geq 2t+1$. Assuming only secure communication channels between each pair of n players (without broadcast), an honest dealer can distribute a secret such that (a) no set of t players has any information about the secret, and (b) any set of players containing at least $t+1$ honest players will be able to reveal the correct secret, with only an exponentially small probability of error.*

We mention some further applications of these results to incomplete communication networks (without broadcast capability). Let G be a synchronous communication network of n processors. In [D], Dolev established necessary and sufficient conditions for the existence of non-cryptographic Byzantine Agreement protocols tolerating up to t Byzantine faults on G. Namely, that $n \geq 3t+1$ and that the connectivity of G is at least $2t+1$. The following theorem extends the fast Byzantine agreement protocols of [FM] to incomplete synchronous networks satisfying Dolev's conditions.

Theorem 2: *Let G be a synchronous network of $n \geq 3t+1$ processors, with connectivity at least $2t+1$, and diameter $d = d(G)$, that provides private and secure communication links between processors that are directly connected by G, then there is an $O(d)$ expected time randomized Byzantine agreement protocol on G tolerating up to t Byzantine faults.*

Similarly, Dolev's conditions are sufficient for general multiparty protocols. Extending the complete network results of [BGW] and [CCD] we have[1]

Theorem 3: *Let G be a communication network between n players satisfying the conditions of theorem 2, then any multiparty protocol can be achieved on G if at most t of the players are faulty. Here again, the secrecy achieved is unconditional, and the probability of error is exponentially small.*

Our main theorem on VSS using the broadcast channel provides the necessary building block for general multiparty protocols under the same conditions. The next theorem provides the most important application of the main theorem, and it provides a strictly stronger result than the previous cryptographic results.

Theorem 4: *Assumming that each participant can broadcast a message to all other participants and that each pair of participants can communicate secretly, any multiparty protocol, or game with incomplete information, can be achieved if a majority of the players are honest.*

Remark: It is important to note that Theorem 4 only guarantees that a set of up to $t < n/2$ faulty players cannot gain any additional information or disrupt the computation. As in all the previous results, it cannot prevent the bad players from sending all the information they are holding to some honest player. Likewise, if among three players one of the players is faulty and stops sending messages, no further secrecy conditions are imposed on the other two honest players left. Therefore, they can exchange all their input information between them and complete the computation. This last example may seem quite puzzling. It is therefore natural to add to the requirements on our protocols that any honest player alone will not hold any additional information (not following from the information in his hands together with the faulty players). This last condition can easily be guaranteed if $n > 2(t+1)+1$ by protecting our computation from coalitions of size $t+1$ instead of t.

[1] See the section on Incomplete Networks for a discussion of stronger results then stated here.

Our results demonstrate the importance of the "Information Checking" idea and we expect this idea to have wide applications. One such possible application is a 1/3-resilient non-cryptographic Asynchronous Byzantine Agreement protocol. Using our information checking we can present such a solution only under a weaker model of adversary, leaving this important question open.

This extended abstract is a combination of two separate papers. The Main Theorem, Theorem 1, and the Information Checking protocol are the main results of Tal Rabin's Master Thesis written under the supervision of Michael Ben-Or [RT]. The other results mentioned above are from [B].

The Model of Computation

In our model of computation we assume a complete synchronous network together with a broadcast channel. The pairwise communication channels between participants are secure, that is, they cannot be read or tampered with by other participants. The broadcast channel enables each participant to send the same message to all the participants, it identifies the sender, and it is completely reliable. In particular, if any participant receives a message via the broadcast channel all other players will receive the same message.

We assume two types of players: *knights* - Honest players or dealers who do not disclose any of the information in their hands, and follow all protocols and, *knaves* - Faulty players or dealers who do as they wish. They may pass incorrect information, or decide to withhold. It is also assumed that all knaves may collaborate and pool their information together. Knaves may have unlimited computing capabilities.

We postulate a *dynamic adversary* who, at any time during a protocol, and having seen all public information and all information possessed by knaves, may choose which additional players will turn into knaves. The total number of players who are knaves is bounded by t. In addition, the dealer may also be a knave.

We choose a security parameter k, so that 2^{-k} is deemed a small enough error margin for our needs. Our protocols will be randomized in the sense that no matter what the adversary and the knaves will do, the probability of an error, i.e., improper outcome, will be smaller than 2^{-k}. All of our protocols and algorithms will be polynomial in the number of players n, and in k.

The Information Checking Protocol

We now proceed to describe a simple process, which will be used later on to solve the Verifiable Secret Sharing Problem. This process enables us to carry out authentication of information. It is not as powerful as Digital Signatures, but on the other hand it needs no cryptographic assumptions.

There are 3 participants in the process: (1) D - dealer, who holds a data s, (2) INT - an intermediary, who is to receive s from D, and who at a later time may wish to pass it on to, (3) R - the recipient.
The player R is said to *accept* s, if he believes that this value originated with D. We wish that the protocol of acceptance will have the following properties:

1. If D and R are knights, then R will always accept s if it actually originated with D, and will reject with probability $\geq 1 - \frac{1}{2^k}$ any other value s'.

2. Regardless if D is a knight or a knave, INT will know, with probability of mistake $\leq \frac{1}{2^k}$, whether R will accept the s that he holds.

For this, a new tool called *Information Checking* is proposed. Information Checking has two major parts called (1) *check vectors* (2) *verification of check vectors*

Information Checking is carried out by the three participants in the following way:
We assume that a large enough prime, $p > 2^k$, has been decided upon for all the computations. Without loss of generality $s \in Z_p$.

Check Vectors:

The dealer D chooses two random numbers $b \neq 0$ and y both in Z_p, and hands to INT the pair (s, y). The dealer D computes $s + by = c$. Then the dealer hands to R the vector (b, c) which will be known as the *Check Vector.*

Later, when INT will transmit (s, y) to the receiver R, the receiver will compute $s + by$ and will accept only if it equals c. It is easily seen that this check vector method essentially disarms a knave INT from the ability to pass to R a false value, s', which R will accept.

Lemma 1 *(1) The probability that INT will deceive R, when the dealer D, is a knight is $\frac{1}{p-1} < \frac{1}{2^k}$.*
(2) The receiver R, has no information about s from his check vector.

Proof of Lemma 1:

1. If INT chooses a new value s', which he would like R to accept he must find the y', which will solve the equation

$$s' + by' = c$$

Only one y' will solve the equation for the b held in R's hand. Thus the probability

$$\Pr(R \text{ will accept } s' \neq s \mid D \text{ is a knight}) = \frac{1}{p-1}$$

2. The receiver R, has no information about s from the check vector that he holds, because all values of s are still possible, with equal probability, due to the fact that for each s there is a single y which satisfies the equation.

Remark Information checking immediately generalizes to the case of several receivers. Let us assume the case where there is one dealer D, one intermediary INT, and receivers $R_1, ..., R_n$. The dealer hands to INT the secret s and random values $y_1, ..., y_n$, y_i will eventually be handed to R_i to authenticate s. Each R_i receives from D a check vector (b_i, c_i) created in the way described above. Even if all R_i's pool their data together , they still have no information about the secret, and for the intermediary who holds the y_i's the probability of deceiving one of the players is $\frac{1}{p-1}$.

Thus the check vector assures the property 1., that if D is a knight, then R accepts the original s transmitted, and rejects substitutes.

Verification of Check Vectors:

To secure property 2 we use the "zero-knowledge" proof technique of [R2]. We modify the above procedure as follows: The dealer transfers to INT an ordered set of pairs $(s, y_1), ..., (s, y_{2k})$ and to R an ordered set of check vectors $(b_1, c_1), ..., (b_{2k}, c_{2k})$, where the y_i and b_i are chosen by D as before and hence

$$s + b_i y_i = c_i \quad for\ 1 \leq i \leq 2k$$

The intermediary INT chooses k distinct random indices $d_1, ..., d_k$ $1 \leq d_i \leq 2k$. He then asks R to reveal $(b_{d_1}, c_{d_1}), ..., (b_{d_k}, c_{d_k})$. For each one of these check vectors, INT computes

$$s + b_{d_i} y_{d_i} = c_{d_i}$$

If *all* k check vectors satisfy the equation, then INT concludes that R will accept the value s that INT holds. Otherwise he concludes that R will reject his value.

We stipulate that R will accept a value s handed over to him by INT, if *any one* of the *unrevealed* check vectors, which he holds for s, satisfies the necessary equation.

Lemma 2 *The probability that the intermediary INT will assume that R will accept s when in fact R will reject it is at most* $\leq 1/\binom{2k}{k} \approx \sqrt{k}/2^{2k}$.

Proof of Lemma 2: If the dealer is a knight then R will never reject s. In order for the error stated in the Lemma to occur, it must be that INT has received k check vectors which are good, and R holds k unrevealed check vectors which are all faulty. Thus, the probability that INT will choose all the k good check vectors is:

$$\Pr(I \text{ assumes } R \text{ will accept } s \mid R \text{ rejects } s) \leq \binom{2k}{k}^{-1}$$

Thus using the protocol of Information Checking as a primitive, we achieve the required 1. and 2.

Secret Sharing when the Dealer is a Knight

In the following protocol we have a group of n players, including at most t knaves, where $n \geq 2t+1$. We assume that the dealer is a knight. He owns a secret s, which he wishes to share among the n players. Without loss of generality we may assume that the secret $s \in Z_p$, for some prime number $p > n$. (The reader should not confuse this new prime p which can be a small prime with the large prime number used for the generation and verification of the check vectors described above.) We want to achieve that any set of fewer than $t+1$ players will have no information about the secret s, and any set of $t+1$ or more players, which contains at least $t+1$ knights, will be able to compute the secret. When the secret is revealed we want that all knights will agree on the same value and that it will be the original secret the dealer shared. This gives a new, simple, polynomial-time, solution for the problem treated in [TW].

Phase 1 - Sharing the secret

1. We fix a prime $p > n$, and n distinct points, $\alpha_1, ..., \alpha_n \in Z_p$, known to all players.
2. The dealer chooses randomly $a_1, ..., a_t \in Z_p$ and defines $f(x) = a_t x^t + ... + a_1 x + s$
3. He computes $f(\alpha_i)$ for all i.
4. For each $f(\alpha_i)$ he chooses random $b_{i1} \neq 0, y_{i1}, ..., b_{in} \neq 0, y_{in}$ and creates the check vectors as described above.

5. The dealer hands over to P_i the data:
   - $f(\alpha_i)$ - P_i's value on the graph of $f(x)$.
   - $y_{i1}, ..., y_{in}$ - P_i will hand y_{ij} to P_j, so P_j can authenticate P_i's piece.
   - $(b_{1i}, c_{1i}), ..., (b_{ni}, c_{ni})$ - check vectors, where (b_{ji}, c_{ji}) is P_i's check vector for P_j's piece.

This completes the dealer's part.
Phase 2 - Revealing the secret When the secret is to be revealed:

1. Every P_i hands to every P_j, $j \neq i$, the pair $(f(\alpha_i), y_{ij})$.
   (Knaves may, of course, give false values.)
2. For all the pairs that P_i received he checks, using (b_{ji}, c_{ji}), that $f(\alpha_j) + b_{ji}y_{ji} = c_{ji}$ and accepts when appropriate. (Each P_i who is himself a knight, will accept at least $t+1$ pieces from the knights, and by Lemma 1 will reject, with high probability, all false pieces from the knaves.)
3. Say P_i accepted $f(\alpha_{i_1}), ..., f(\alpha_{i_r})$, $r \geq t+1$. He can now take $t+1$ pieces and compute $f(x)$ by interpolation. Since the dealer was a knight *all* the accepted points lie on the graph of $f(x)$. The reconstructed secret s is uniquely defined as the constant term of $f(x)$, and equals the one initially distributed.

It is easy to see that the above algorithm provides us with the possibility for a knight dealer to share a secret, so that the knaves can not interfere in the computations, and that all knights will compute the same value s. This completes the proof of Theorem 1.

The Weak Secret Sharing - WSS

For the next two problems we add to our model the option of *broadcasting*. Each player can announce a message which will be heard by all others. This is necessary for $2t+1 \leq n \leq 3t$, otherwise Byzantine Agreement can be used.

Definition 1: It will be said that players $P_1, ..., P_r$, $r \geq t+1$, holding pieces given by a dealer, *define a secret* s, if (1) every player P_i has a distinct evaluation point. (2) for every subset $P_{i_1}, ..., P_{i_{t+1}}$, the polynomial $f(x)$ of degree t, interpolated through the pieces $\beta_{i_1}, ..., \beta_{i_{t+1}}$, which they hold, is the same, and its free term is s.
Note: Checking whether a set of players $P_1, .., P_r$ define a secret s, is polynomial.

Under this model with broadcast, we shall define the following properties for *Weak Secret Sharing*:

1. The properties of the previous secret sharing protocol, when the dealer is a knight.
2. If D is a knave then after it has been decided, (in a sense to be explained later on), that a secret has been distributed, either
   - *All* knights will agree upon a secret s which they define (Definition 1) and which is the one shared to *them* by the dealer, or
   - *All* knights decide to disqualify the dealer.

WSS-Theorem: *Assume $n \geq 2t+1$. For a dealer D, who is present until the end of the protocol, there is a protocol which achieves the Weak Secret Sharing. Moreover, the protocol is of polynomial complexity in both n and the security parameter k, with any desired probability of success.*

Weak Secret Sharing Protocol. This protocol is executed in three phases. The first two phases are the ones which ensure that when the secret will be revealed, the desired properties will be achieved. The last phase is the one in which the secret is revealed.
Phase 1 : Sharing the secret

1. Following steps 1-4 above, the dealer shares the secret using a random polynomial and creates check vectors to all the pieces.
2. He hands over to P_i the data:
   - $\beta_i = f(\alpha_i)$ - P_i's point on the graph of $f(x)$.
   - $y_{i1}, ..., y_{in}$ - P_i will hand y_{ij} over to P_j, so P_j can authenticate β_i.
   - $(b_{1i}, c_{1i}), ..., (b_{ni}, c_{ni})$ - check vectors, where (b_{ji}, c_{ji}) is P_i's check vector for β_j.

This is precisely the protocol executed by a dealer who is a knight. But when D is a knave he can hand over to P_i arbitrary values for $\beta_i, y_{i1}, ..., y_{in}, (b_{1i}, c_{1i}), ..., (b_{ni}, c_{ni})$.

Note, whenever a check vector (b_{ji}, c_{ji}) for the value β_j, is mentioned in the above, it is not a single check vector but a set of $2k$ check vectors for β_j. The y_{ij} is also a set of $2k$ random values for authentication of β_i. All these are required for the verification of check vectors as described in Information Checking.
Phase 2: Modified verification of the check vectors
The following protocol is carried out by each P_i with respect to every other player P_j. The player P_i is to be the intermediary as defined in the Information Checking Protocol, and P_j will be the receiver. The dealer is the dealer who shared the secret. P_i wants to verify the check vectors held by all other players

for his piece β_i. The modification is that if P_i is not satisfied that all players will accept his piece, he will take measures to ensure that they will.

1. The intermediary P_i, chooses k check vectors in the manner described in Information Checking, which he then asks P_j to *broadcast.*

2. The receiver P_j broadcasts the requested k check vectors.

3. The dealer may: (1) broadcast an approval of these check vectors, or (2) create a new check vector for β_i in the usual manner, hand over to the intermediary P_i the number y_{ij}, and *broadcast* (b_{ij}, c_{ij}). This broadcasted check vector is now taken to be P_j's check vector for β_i.

4. If the first instance in 3. has occurred, P_i will continue the verification of check vectors as described above. If he knows that P_j will accept his point he moves on to the next player to carry out this protocol with him. If he decides that P_j will reject β_i, he broadcasts a request that the dealer D broadcast his piece on the graph $f(x)$. After P_i's piece has been broadcasted he will stop the verification of check vectors.
   If the second instance in 3. has occurred, P_i checks the single check vector, broadcasted by the dealer, and according to the result takes the same measures as defined before.

Lemma 3 *In this modified verification of check vectors, data unknown to knaves is never exposed.*

Proof of Lemma 3: Data unknown to knaves is data which is transferred between two knights. If the dealer exposes P_j's check vectors it is because D and/or P_j are knaves, otherwise D will be content with what P_j had broadcasted. If INT is a knight and he requests the disclosure of his piece, then it must be that D is a knave. This is because the dealer has the last say about the check vectors and if after that they are still faulty, then he must be a knave - so the information broadcasted is the dealer's information which is already known to the knaves.

Remark The above verification scheme assures a knight P_i, acting as intermediary, that his piece will be accepted by P_j, the receiver, or that the piece associated with him has been made public by the dealer during the process.
It is not important if the piece γ_i exposed by D differs from β_i, the piece held by P_i, because γ_i is now a value to which D has committed himself, as being P_i's point on the graph $f(x)$.

At this point is is useful to introduce the following
Definition 2: A *coalition* is a subset $C \subseteq \{P_1, ..., P_n\}$ of the set of all players, for which at the end of Phase 2: each $P_i \in C$ knows that every other player $P_j \in C$ will accept his piece of the shared secret.

Note that there may be many coalitions among players, and that one player may be a member in several coalitions. However, the following lemma holds

Lemma 4 *At the end of the verification of check vectors (Phase 2) the set of players who are knights is a coalition.*

Proof: If player P_i is a knight, he was either satisfied that all players will accept his point, including all knights, or if he thought that one player will not accept his point ,then he requested that it be broadcasted. In the latter case his piece will definitely be accepted by all players. We see that P_i's piece will in fact be accepted by all players. This is true for every player P_j who is a knight - thus the knights form a coalition.

Phase 3 - Revealing the secret

1. The dealer D broadcasts his polynomial $f(x)$, which defines the secret s.

2. Each P_i hands to P_j his piece β_i and the corresponding y_{ij}'s.

3. P_i checks, by use of his check vectors, for every pair (β_j, y_{ji}) if he accepts the piece, and creates the list $\beta_{i_1}, ..., \beta_{i_r}$ of all pieces which he has accepted. (For each knight it is known that $r \geq t+1$ because of the fact that the set of all knights is a coalition).

4. For each piece β_i that P_i accepted, he checks to see if it is on the graph $f(x)$. If some piece does not fit $f(x)$, P_i broadcasts his vote to disqualify the dealer D.

5. The dealer is disqualified if at least $t+1$ players so voted.

6. If the dealer was not disqualified, s, the free term of $f(x)$, is taken to be D's secret.

Proof of the WSS Theorem:

1. When the dealer is a knight -
   We need to show that he will not be disqualified, and that all knights will agree on the same secret s, results immediately from the properties of the Information Checking.

2. When the dealer is a knave -
   It is left to show that if he was not disqualified, then the secret s, decided upon is the secret defined by all knights.
   Step 4 of Phase 3, and the coalition of the knights ensures this. Let us say P_i is a knight and his piece is not on $f(x)$, then he casts his vote against the dealer. Because all other knights accept his piece, due to the coalition among them, they will also cast the same vote, and the knave dealer will be disqualified. Now if they agree on $f(x)$, it must be that they define it, (see Definition 1), thus it is proper to take s, the free term of $f(x)$, to be the secret.

The Verifiable Secret Sharing - VSS

We now turn to the proof of the main theorem of this paper.

Main Theorem: (Verifiable Secret Sharing)
Assume $n \geq 2t+1$, at most t players who are knaves, and a dealer D, who may be a knight or a knave. There exists a protocol which achieves the Verifiable Secret Sharing. Moreover, the protocol is of polynomial complexity in both n and the security parameter k, and guarantees any desired probability of success by an appropriate choice of k.

We start by stating the protocol. The protocol has 4 phases: sharing the secret by the dealer; a second-level secret sharing by the P_i's; zero knowledge proof; and computing the secret. At the end of the third phase it will be said that a proper secret has been shared by D.

Phase 1 - Sharing the secret

1. The dealer chooses randomly $a_1, ..., a_t \in Z_p$ and defines $f(x) = a_t x^t + ... + a_1 x + s$
2. He also picks random polynomials, $g_1(x), ..., g_{k \cdot n}(x)$, each of degree t.
3. He computes $f(\alpha_i)$, $g_j(\alpha_i)$ *for* $1 \leq i \leq n$, $1 \leq j \leq k \cdot n$
4. He hands over to P_i the data:
   - $\beta_i = f(\alpha_i)$ - P_i's point on the graph of $f(x)$.
   - $\gamma_{ji} = g_j(\alpha_i)$ for all j's - P_i's points on the polynomials $g_j(x)$.

This is the protocol executed by a dealer who is a knight. But when D is a knave he may of course hand over to P_i arbitrary values for $\beta_i, \gamma_{1i}, ..., \gamma_{k \cdot n,i}$

Phase 2 - Second level secret sharing

Each P_i shares by WSS, each piece received from the dealer D, i.e., the pieces $\beta_i, \gamma_{1i}, ..., \gamma_{k \cdot n,i}$ Let us say that P_i shares β_i using $h_i(x)$, and γ_{ij} using $h_{ji}(x)$. Player P_i also computes

$$\delta_{ji} = \beta_i + \gamma_{ji}, \ 1 \leq j \leq k \cdot n,$$

and creates and distributes check vectors for these values. He does not share the value δ_{ji} by a separate, independent, random, polynomial, because the sum of the two polynomials $h_i(x) + h_{ji}(x)$ which shared β_i and γ_{ji} respectively, has as its free term the value δ_{ji}.

Phase 3 - Zero Knowledge Proof

This phase will ensure that D has in fact shared a secret s, among the players. In this phase it is possible for players to be disqualified. The player P_i can be *disqualified* if:

1. the secret he shared using the WSS disqualifies him, or
2. a vote of at least $t+1$ players is cast against him.

The protocol for this phase:
For each $g_j(x)$, player $P_{j \bmod n}$, $1 \leq j \leq k \cdot n$, decides whether D will expose $g_j(x)$ or $f(x) + g_j(x)$. (This procedure of choosing the polynomial ensures that at least $k(t+1)$ of the alternatives are randomly chosen). In the j-th round we set $H(x)$ to be $g_j(x)$ or $f(x) + g_j(x)$ depending on the player's decision. This version of zero-knowledge proof in the context of VSS was used in [CCD], when $n \geq 3t + 1$.

1. The dealer broadcasts $H(x)$.
2. Each P_i checks if his piece γ_{ji}, or the sum of his two pieces $\delta_{ji} = \beta_i + \gamma_{ji}$, as the case may be, is on $H(x)$. If P_i is not satisfied then he requests that the dealer D expose *all* the information which he had handed to P_i in Phase 1.
3. If the dealer D, does not broadcast the required information, the knights decide that *no* secret was shared.
4. Every P_i whose pieces were not broadcasted by D in 2, now broadcasts his piece (γ_{ji} *or* δ_{ji}, as the case may be) and the polynomial by which he shared his piece using the WSS. Because P_i used the WSS protocol he may now
   - Be disqualified, or
   - Expose a piece (γ_{ji} *or* δ_{ji}) on which all knights agree.

   All players check if this piece is on $H(x)$, if yes then OK. Otherwise P_i is disqualified, because at least $t + 1$ knights will cast a vote against him.

5. If at any time $t+1$ pieces of the secret have been made public, the polynomial which they define is taken as the dealer's secret s. At each stage of the computation the validity of all public points is checked. If one does not match a certain polynomial publicized by D, then it is decided that *no* secret was shared.

Lemma 5 *In step 2 data unknown to the knaves is never revealed.*

Proof of Lemma 5: If P_i is a knave - trivial.
If P_i is a knight then the only instance in which his piece γ_{ji} or δ_{ji} does not fit $H(x)$ (and hence causes him to request disclosure), is when D is a knave, because D is the one who handed P_i the pieces β_i and γ_{ji}.

Lemma 6 *A knight player can never be disqualified.*

Proof of Lemma 6: A knight can not be disqualified by his WSS as proven in Theorem 1. And he will not be disqualified if his piece is not on $H(x)$, because he would have complained about it in step 2.

Lemma 7 *At the end of Phase 3 with probability $2^{-k\cdot(t+1)}$, all pieces held by non-disqualified players and all public pieces define the same polynomial of degree t.*

Proof of Lemma 7: If a player P_i has shared a piece $\beta_i \neq f(\alpha_i)$, then for all j only *one* of the following can be true:

1. $\gamma_{ji} = g_j(\alpha_i)$
2. $\delta_{ji} = f(\alpha_i) + g_j(\alpha_i)$

If 1. is true, $\beta_i + \gamma_{ji} = \delta_{ji} \neq f(\alpha_i) + \gamma_{ji}$.
If 2. holds then $\beta_i + \gamma_{ji} = \delta_{ji} = f(\alpha_i) + g_j(\alpha_i)$ but $\beta_i \neq f(\alpha_i)$ implying that $\gamma_{ji} \neq g_j(\alpha_i)$.
Because the decision whether to disclose $g_j(x)$ or $f(x) + g_j(x)$ is made randomly (by the $t+1$ knights) at least $k \cdot (t+1)$ times, we achieve the desired probability of being correct.

Phase 4 - Computing the dealer's secret

1. All non-disqualified P_i's broadcast their piece β_i and the polynomial $h_i(x)$, which was used in the WSS for that piece.
2. The β_i's and all public pieces form a set - S.
3. For each piece β_i in S (excluding public pieces), the non disqualified players check that it was in fact the one shared by $h_i(x)$ in the WSS. If a player P_i who holds the piece β_i is disqualified due to the WSS, then the piece β_i is taken out of S. After this it holds that $|S| \geq t+1$, because at least all the knights' pieces are in S, due to Secret Sharing when the Dealer is a Knight.
4. Each player may take any $t+1$ pieces from S and interpolate the polynomial $f(x)$. The secret s, will be the free term of the polynomial $f(x)$.

Proof of the Main Theorem: The proof will be divided into two parts:

1. We will prove that if the dealer is a knight then it will never be decided that *no* secret was shared.
2. For any dealer, if Phase 3 ended with the decision that a secret has been shared, then all knights will agree to the same secret s, and it will be the one defined by the pieces which they hold and the public pieces.

We now proceed to prove the above:

1. The decision that no secret was shared arises if
   - the dealer refuses to broadcast information requested by one of the players, or
   - public pieces which he broadcasted do not fit on a polynomial which he broadcasted

   Since the dealer is a knight, both instances are impossible.
2. At the end of Phase 3, when all knights know that a proper secret has been shared, it is also known, by Lemma 7, that all pieces held by non-disqualified players, and all public pieces define the same polynomial. Due to the fact that each one of the unpublicized pieces is shared using WSS, in Phase 4 it must be either revealed *properly* or the dealer of that piece will be disqualified. But a knave will not be able to introduce a new piece into the computations, which will be accepted by the knights, (Theorem 3.1). All points revealed by knights will be accepted (Secret Sharing when the Dealer is a Knight). So we will have at least $t+1$ pieces which define the same polynomial. Thus the desired properties are achieved.

Remark: Instead of revealing the secret to all the players we can reveal the secret to only one of the players by having all the information sent only to this player.

Incomplete Networks

Let G be a synchronous network of $n \geq 3t+1$ processors, with connectivity at least $2t+1$, and diameter $d = d(G)$, that provides private and secure communication links between processors that are directly connected by G. To prove both Theorems 2 and 3, it is enough to show how to simulate in our partial network a secure communication line between any two honest players given that they are connected by $2t+1$ vertex disjoint paths in $O(d)$ steps. Having such a scheme we can simulate on the network G the constant expected time Byzantine Agreement protocol of Feldman and Micali [FM] in $O(d)$ expected time, proving Theorem 2, or simulate the secure computation protocol of [BGW], proving Theorem 3.

We can therefore assume that the sender of the message D is honest, and so we are in a case where Theorem 1 applies. To send a message to R, D encodes the message using a polynomial of degree t into $2t+1$ pieces, and sends each piece together with check vectors for all the other pieces along a separate route to R. To read the message, R tests each piece with all the check vectors arriving from all the routes and discards any piece that does not match with at least $t+1$ such vectors. Having collected $t+1$ such pieces, R computes the interpolation polynomial and takes its free term as D's message. It is easy to see this simple *one phase* protocol has all the properties needed to prove Theorem 2 and Theorem 3.

After submitting this abstract, the authors have learned that Dolev, Dwork and Yung [DDY] have shown how to simulate in a $2t+1$-connected network a secure communication line between any two honest players in a completely error free manner. Therefore our Theorem 3 is not the best possible, as under the conditions of Theorem 3, any multiparty protocol can be achieved with absolutely no error at all (see [DDY]). We note that in the "error-free" simulation, the faulty players can delay the message between two honest players for $O(td)$ steps. Therefore their simulation does not provide a proof of Theorem 2. We briefly sketch below how to combine both results and obtain an absolutely error free secure transmission in $O(d)$ expected time.

Error Free transmission

Assume D and R are $2t+1$ connected, and D wants to transmit the message M to R. D randomly writes $M = M_1 + M_2$, and sends using the scheme described above M_1 to R. R collects all the verified messages (i.e. at least $t+1$ verifications), and if there is a polynomial of degree t that interpolates through *all* the points, R takes the free coefficient as M_1. In this case R broadcasts an acknowledgement on all the $2t+1$ lines to D, and D tries to send M_2.

On the other hand, if R cannot complete the interpolation through all the points, R broadcasts back to D (along the $2t+1$ lines) all the information he received from all lines. This step does not relay any information because this will be done either for M_1 or for M_2 but never for both, and clearly M_1 ($M2$) alone contains no information.

At this stage, D compares the broadcast of R to its original messages. D marks all the miss-matchs as "bad", and broadcasts the "bad" list to R. Concurrently, D breaks up the message M into new random pieces $M = N_1 + N_2$, and begins the whole procedure again, but now R will ignore the shares of the secret coming from connections on the bad list.

We continue (using on each retry a new breakup of M) until either:
(a) Both pieces arrive in a good shape at R, or
(b) The "bad" list is longer than $t+1$, or
(c) after a bad transmission the bad list doesn't grow.

Clearly, after at most $t+1$ retrys, a pair of good D and R will be able to transmit their value securely with no errors at all. Furthermore if only one of D and R is bad, the good player will never wait for more than $t+1$ rounds. Moreover, because of the check vectors, only with exponentially small probability there will be any need for retransmission.

Note: The procedure described above, without the check vectors, is only a slight variation of the transmission scheme of [DDY]. It is clear that this procedure works as well without the check vectors, but there is a clear advantage in using the check vectors during this procedure. Without the check vectors, the t bad players can cause a delay of $O(t \cdot d)$ steps in the communication between good players, where as using the check vectors, only with exponentially small probability there will be any need for retransmission.

It is clear that if D and R are both good, the expected number of communication rounds to get an error free transmission is constant ($O(d)$). A slightly more complicated problem is to handle the case of a bad D sending a message to a good R. In this case using the scheme described above, the bad D can easily cause a delay of $O(td)$ time till R will be able to continue. Thus unless we are willing to give up the synchrony of our simulated protocol a single bad player can cause a delay of $O(td)$ steps.

To avoid this problem we make D the dealer of the secret in our VSS protocol. Assuming R is good we get a broadcast channel simulation between all the $2t+1$ connections and the dealer D as follows:

- A broadcast of one of the $2t+1$ is sent to R who sends it on all $2t+1$ connections, and they transmit the message also to D (who gets the message by a simple majority rule).
- A broadcast from D to all the players is done by having D send the message on all lines to R. R gets the message (by majority) and sends the value he got to all the $2t+1$ connections. This value, coming from R is taken as the value of D's broadcast.
- The private line between two connections can go through R.

Since the VSS uses a constant number of rounds so will this simulation. Since the VSS guarantees with exponentially small probability of error that there will be a t-degree polynomial interpolating through all the accepted points, R will get a message (or realize that D is faulty) in constant number of rounds with only an exponentially small probability of not being able to decide what happened. In this case we can go back to the previous scheme, and continue for at most $O(t)$ more rounds. Since this happens only with exponentially small probability, we get an $O(d)$ expected time transmission even from a bad D to a good R. (The case of faulty R and good D is not needed at all).

From this constant expected time error-free transmission we can prove a stronger version of Theorem 3.

Theorem 3*: *Let G be a synchronous network of $n \geq 3t+1$ processors, with connectivity at least $2t+1$, and diameter $d = d(G)$, that provides private and secure communication links between processors that are directly connected by G, then any multiparty protocol $\mathcal{P}$ can be achieved on G if at most t of the players are faulty. As in [DDY], the computation is completely error-free, and the secrecy achieved is unconditional. Moreover, the expected computation time is only $O(d)$ times the best secure and error-free protocol for $\mathcal{P}$ on a complete network.*

Multi Party Computations

In this section, all computations are carried out in a small finite field, for example Z_p where $n < p < 2n$. Our VSS protocol can be easily modified to work in a small finite field or by using a separate large prime for the check vectors.

We say that a group of players $P_1, ..., P_n$ *will have computed a function $f(u_1, ..., u_m) = u$ by a protocol PR*, where $u_1, ..., u_m$ were entered into the computation using VSS, if at the end of the protocol, u is a verified shared secret.

The problem of multiparty computations can be reduced, (see [GMW]) to the problem of multiparty function computations. We will produce protocols for addition and multiplication of shared secrets and for multiplication by a publicly known constant, which can be carried out in a fault tolerant and secrecy preserving manner. Thus we achieve that a group of n players can secretly and with fault tolerance compute any function.

Definition - A group of n players holds a *verified secret (data) s*, shared using the polynomial $f(x)$, so that $f(0) = s$, and satisfying the conditions of **VSS** if:

1. The polynomial $f(x)$ is of degree t.
2. Each player, P_i, holds a share of the secret $\beta_i = f(\alpha_i)$.
3. Every piece β_i was shared by P_i, using WSS

Note - A result of using our VSS protocol to share a secret s (Theorem), is that s will be a *verified secret.*

Now, all we need to show, in order to prove Theorem number, is that we can produce protocols for addition and multiplication, of verifiable shared secrets u and v, which maintain the situation that all data in the computation, including the end result, is verified data.

Let us look at two secrets u and v, which are verified secrets. Thus, it is known that:

1. They were shared by $f^u(x)$ and $f^v(x)$ respectively, both polynomials of degree t.
2. Each P_i holds:
   - A piece $f^u(\alpha_i) = \beta_i^u$, which he further shared using the polynomial $h_i^u(x)$ and the WSS protocol.
   - A piece $f^v(\alpha_i) = \beta_i^v$, which he further shared using the polynomial $h_i^v(x)$ and the WSS protocol.
   - The pieces $h_j^u(\alpha_i) = \beta_{ji}^u$ and $h_j^v(\alpha_i) = \beta_{ji}^v$, which are his shares of the other players' pieces. (The player P_i received them as a result of WSS).

Addition Protocol

We would like to compute $u+v$, and to achieve that the sum will be a verified secret.
The polynomial $f^u(x) + f^v(x) = f^{u+v}(x)$ has as its

constant term the desired value $u+v$, and this polynomial is of degree t. If each P_i adds his pieces $\beta_i^u + \beta_i^v = \beta_i^{u+v}$, then he will have a share of $u+v$ by the polynomial $f^{u+v}(x)$. Therefore, conditions 1 and 2 for a verified secret hold immediately.

To achieve condition 3, we need that β_i^{u+v} will be shared using WSS.
Note that the value β_i^{u+v} is already shared by the polynomial $h_i^{u+v}(x)$, which is the sum of the polynomials which shared β_i^u and β_i^v, that is $h_i^u(x)+h_i^v(x) = h_i^{u+v}(x)$. Thus all we need are the appropriate check vectors. This is achieved as follows:

1. The player P_i creates check vectors for $h_i^u(\alpha_j) + h_i^v(\alpha_j) = h_i^{u+v}(\alpha_j)$, and distributes them to the other players.

2. All players verify these check vectors using the Modified Verification of Check Vectors Protocol (see Section).

During the computation of the addition, knaves may act as they wish, but if they don't follow the protocol they are disqualified. (Their data can be reconstructed if needed in a manner which will be described later).

Lemma 8 *After the above addition protocol is carried out, the sum $u+v$ is a verified secret.*

Note - The same protocol can be carried out for subtraction.

Protocol for Multiplication by a Constant

There is a publicly known constant d, and we would like to compute the value $u*d$ so that this product will be a verified secret. This multiplication is trivial. All that P_i needs to do is multiply his share β_i^u by the constant d. When at a later stage he receives the piece β_j^{u*d} and the value y_{ji}^u from player P_j, he should use the check vector (b,c), which he holds for the original piece of the player P_j, by computing, $\beta_j^{u*d} + bdy_{ji}^u = cd$.

Linear Computations

The combination of the protocol for addition and the protocol for multiplication by a constant, provides us with the ability to compute any linear combination of verified secrets.

Multiplication Protocol

Before we proceed to show how to compute the product of two verified secrets, let us prove two lemmas which will be needed later.

Lemma 9 *A dealer can prove that two secrets, s_1 and s_2, which he shared using VSS, are equal.*

Proof: All players carry out the subtraction protocol for s_1 and s_2, Based on the lemma for addition/subtraction this new value is a verified secret, and can be reconstructed by the players. They reconstruct the value $s_1 - s_2$. Now, $s_1 - s_2 = 0$ iff $s_1 = s_2$.

Lemma 10 *Using the Zero Knowledge technique the dealer can prove that three secrets, a, b and c, satisfy $a \cdot b = c$.*

Proof: Let f, g and h be the polynomials used to distribute a, b and c respectively. We recall that all our secrets are elements of a small finite field E, $|E| = p$, and we can pick p to satisfy $n < p < 2n$. Let $Z = \{(x,y,z)|xy = z, \text{ for all } x,y \in E\}$ be the multiplication table of E. The dealer randomly permutes the members of Z to a list $\{(x_i,y_i,z_i)\}$. He then generates random polynomials of degree t, f_i, g_i and h_i, $i = 1 \ldots p^2$, encoding this list, (e.g. $f_i(0) = x_i$, etc.). The dealer prepares $m = knp^2$ independent encoded copies of the multiplication table of E and distributes all the randomly chosen triples in these lists using our secret sharing protocol. At this point each player broadcasts a list of kp^2 requests asking the dealer to either open completely one of the encoded lists or to show, using the equality protocol of Lemma 9, that for some i on the list $f(0) = f_i(0)$, $g(0) = g_i(0)$ and $h(0) = h_i(0)$. It is easy to see that this procedure provides a zero knowledge proof to the fact that $h(0) = f(0)g(0)$, with probability of failure at most 2^{-k}.

We now proceed to exhibit the multiplication protocol. For the above u and v, we would like to compute $u*v$, and to ensure that the product will be a verified secret.

The polynomial $f^u(x) * f^v(x) = f_1^{u*v}(x)$ has as its constant term the desired value $u*v$, but two problems arise: (1) The polynomial is of degree $2t$. (2) It is not a random polynomial. To overcome these problems we will adapt the methods described in [BGW], and in combination with the VSS protocol we will return to the state in which all data in the computation is verified data. The protocol is as follows:

1. The player P_i must extend his WSS of β_i^u and β_i^v to VSS. This is easily done: (the protocol given is for β_i^u, and the same is done for β_i^v).
   - Each P_j holds a piece β_{ij}^u which is a share of P_i's β_i^u. He shares β_{ij}^u using WSS.
   - Then P_i carries out the zero knowledge proof, as in our VSS protocol, to show that all shares held by the players lie on a polynomial $h_i^u(x)$ of degree t.
2. If some player refuses to cooperate, or is disqualified in the above process, then his pieces are reconstructed by the other players, and become common knowledge. The players which participate in this computation, are players who have extended their WSS to VSS, that is that they were not disqualified. Thus, their pieces β_j^u, which are now shared by VSS, are points on the graph of the polynomial of the shared secret u. Assume that P_i didn't follow the protocol, his piece β_i^u is computed in the following way:

   Since β_i^u is on a polynomial of degree t, we can represent β_i^u as a linear combination with publicly known (and easily computable) coefficients of any other $t+1$ point on this polynomial. Since at least $t+1$ players will extend the WSS of their points on this polynomial to a VSS, we can use this linear relation together with the Linear Computation Lemma to compute a verified secret of the missing β-s and make them public.

   Now it holds, that either β_i^u was shared using VSS, or it was made public.
3. The player P_i computes $\beta_i^u * \beta_i^v = \beta_i^{u*v}$, and distributes it with a new polynomial $h_i^{u*v}(x)$, using the VSS protocol. Then he proves that $h_i^u(0) * h_i^v(0) = h_i^{u*v}(0)$, as described above in Lemma 10. If he fails to do the above, his pieces are reconstructed and exposed.
4. In order to randomize the polynomial $f_1^{u*v}(x)$, each P_i shares t random polynomials $g_{i,j}(x)$, of degree t, and proves that he has done just this. Using the method described in [BGW], they generate a random polynomial of degree $2t$ with a zero free coefficient. By adding this polynomial to $f_1^{u*v}(x)$ we obtain a new random polynomial $f_2^{u*v}(x)$ with the same free coefficient.
5. To preform the reduction of the degree of the polynomial, $f_2^{u*v}(x)$ to degree t, we carry out the linear computation described in [BGW]. If a player doesn't cooperate, his piece is revealed using the VSS for this piece. The result of this computation will be that each player holds a verified share of a piece of $u*v$, shared by the polynomial $f^{u*v}(x)$ of degree t, so that $f^{u*v}(0) = f_1^{u*v}(0)$. Each player manages to compute his own, new piece, after all information held by the other players has been passed on to him.
6. As a result of the above computation the following facts hold:
   - Each player holds a share β_i^{u*v} of the secret $u*v$.
   - This piece is already shared using VSS.
   - Due to the VSS of β_i^{u*v}, each player P_j holds a share β_{ij}^{u*v} of β_i^{u*v}.

   To achieve condition 3 of a verified secret for $u*v$, β_i^{u*v} must be shared by WSS. To transform the VSS of this piece to WSS the following is done:
   - The player P_i creates and distributes check vectors for all β_{ij}^{u*v}.
   - All players execute the Modified Verification of Check Vectors.
   - If P_i fails to cooperate his piece is exposed using the VSS for it.

   We therefore have

Lemma 11 *After the above multiplication protocol is carried out the product $u*v$ is a verified secret.*

And this completes the proof of Theorem 4.

Acknowledgement

The first author wishes to thank her advisor Michael Ben-Or for suggesting the VSS problem and for his encouragement and many important comments, as well as Michael Rabin for several useful comments.

References

[B] M. Ben-Or, Multiparty Protocols with Honest Majority, in preparation.

[BGW] M. Ben-Or, S. Goldwasser and A. Wigderson, Completeness Theorems for Non-Cryptographic Fault-Tolerant Distributed Computation, STOC88, pp. 1-10.

[CCD] D. Chaum, C. Crepeau and I. Damgard, Multiparty Unconditionally Secure Protocols, STOC88, pp. 11-19.

[CGMA] B. Chor, S. Goldwasser, S. Micali and B. Awerbuch, Verifiable Secret Sharing and Achieving Simultaneity in the Presence of Faults, FOCS85, pp. 383-395

[D] D. Dolev, The Byzantine Generals Strike Again, J. of Algorithms, Vol. 3, pp. 14-30, 1980.

[DDY] D. Dolev, C. Dwork, M. Yung, NonCryptographic Secure Communication in General Networks. Preprint.

[F] P. Feldman, Optimal Algorithms for Byzantine Agreement, MIT Ph.D. Thesis, to appear.

[FM] P. Feldman, S. Micali, Optimal Algorithms for Byzantine Agreement, STOC88, pp. 148-161.

[GMW] O. Goldreich, S. Micali and A. Wigderson, How to Play Any Mental Game, STOC87, pp. 218-229.

[R1] M. Rabin, Randomized Byzantine Generals, FOCS83, pp. 403-409

[R2] M. Rabin, Digitalized Signatures, Foundations of Secure Computations, R. Demillo et.al. editors, Academic Press, (1978), pp. 155-165

[RT] T. Rabin, Robust Sharing of Secrets when the Dealer is Honest or Cheating, M.Sc. Thesis, The Hebrew University, July 1988.

[S] A. Shamir, How to Share a Secret, CACM, 22, pp. 612-613, (1979)

[TW] M. Tompa and H. Woll, How to Share a Secret with Cheaters, IBM Research Report, RC 11840 (Log # 52910) (1986)