Indian Statistical Institute Kolkata

M.Tech (Computer Science Dissertation)

Studies on Verifiable Secret Sharing

Author:
YOGRAJ SINGH
Roll No:CS0920

Supervisor:
Dr. K.C. GUPTA
Applied Statistics Unit

A dissertation submitted in partial fulfillment of the requirements for the award of M.Tech.(Computer Science) degree

Indian Statistical Institute

Certificate of Approval

This is to certify that the thesis entitled "Studies on Verifiable Secret Sharing " submitted by **Yograj Singh** towards partial fulfilment for the degree of M.Tech. in Computer Science at Indian Statistical Institute, Kolkata, is a record of the work carried out by him under my supervision and guidance.

Date:
Kolkata

(Dr. K.C.Gupta)
Applied Statistics
Unit
Indian Statistical
Institute

Acknowledgements

It is a pleasure to convey my gratitude, to few of the many sources of inspiration for me, to all the people who have helped in successful realization of this dissertation. First and foremost I thank the incessant source of divine blessings, the almighty God, who always motivates me to move forward with his omens and love.

I express my deep sense of gratitude to my supervisor **Dr. Kishan Chand Gupta**, Assistant Professor, Applied Statistics Unit, Indian Statistical Institute Kolkata, India and **Dr. Ashish Choudhury**, Indian Statistical Institute, Kolkata. Thank you so much Ashish Sir, and K. C. Gupta sir! I feel privileged to express my sincere regards to my guide and Ashish Sir for their valuable guidance, support and constant encouragement throughout the course of my project work. Their truly scientific intuition and broad vision inspired and enriched my growth as a student. The critical comments, rendered by them during the discussions are deeply acknowledged.

I gratefully acknowledge Shashank and all the people who have helped me in any way during my M.Tech course work. I am indebted to the Department of Computer Science, Indian Statistical Institute, Kolkata, to all its faculty and staff for their academic support and encouragement. I feel a deep sense of gratitude for my family who raised me with their unconditional love and taught me to always believe in myself.

Finally, I would also like to thank all the readers of this work, since any piece of academia is useful if it is read and understood by others so that it can become a bridge for further research.

Abstract

In this Thesis we will discuss three important tools in cryptography namely, *Secret Sharing, Verifiable Secret Sharing* and *Weak Secret Sharing.* In all secure systems that use cryptography in practice, keys have to be protected by encryption under other keys when they are stored in a physically insecure location. But the keys used for protection have to be protected themselves, so no matter what we do, we cannot avoid having one or more keys in our system that are only protected because they are stored in a physically secure way. These are typically very high priority keys, such as the secret key that a certification authority (CA) uses to create certificates. Precisely because such a key is so important, it would be a disaster if it was revealed to an adversary. But it would be equally bad if the key was lost and could not be retrieved. In other words, there is a big need to keep such keys secret and available at the same time.

This seemingly puts designers of security systems in a rather difficult dilemma: to make sure that a key is not revealed to anyone, one is inclined to store it only in a single, very secure location; while the need to make sure the key is always available seems to imply that you should store the key in as many different locations as possible. *Secret sharing* is a technique that allows us to nevertheless address both of these concerns at the same time.

Alike other methods secret sharing also have some limitations. To over come the shortcomings of secret sharing the notation of *Verifiable secret sharing (VSS)* is introduced. VSS is a two phase protocol (Sharing and Reconstruction) carried out among n players in the presence of an adversary who can corrupt up to t players. The goal of the VSS protocol is to share a secret S, among the n players during the sharing phase, such that in the reconstruction phase, the secret is reconstructed correctly. In this thesis we will discuss some known VSS protocols under the *information theoretic setting* over *synchronous* network, considering an *active unbounded adversary. Weak secret sharing* (WSS) is a variant of verifiable secret sharing, where the reconstructed value may also be some default value, in case the dealer is corrupted.

In the present thesis we have proposed a 1-round WSS protocol having *communication complexity of* $O(n)$. Previously, a 1-round WSS protocol with *communication complexity* of $O(n^2)$ was proposed. Thus, we get an improvement of $\theta(n)$ in the *communication complexity.*

Contents

Chapter 1

Secret Sharing

1.1 Motivating Examples:

(1) Suppose you and your friend accidentally discovered a map that you believe would lead you to an island full of treasure. You and your friend are very excited and would like to go home and get ready for the exciting journey to the great fortune. **Now who is going to keep the map?** Suppose you and your so-called friend do not really trust each other and are afraid that, if the other one has the map, he/she might just go alone and take everything. Now we need a scheme that could make sure that the map is shared in a way so that no one would be left out in this trip. **What would be the Solution?**
A possible solution is to split the map into two pieces and make sure that both the pieces are needed in order to find the island. You can happily go home and are assured that your friend has to go with you in order to find the island. This illustrates the basic concept of *secret sharing.* In this example, map is the secret, while you and your friends are two parties that share the secret.

(2) Imagine that you have invented a new, burger sauce that is even more tasteless than your competitors. This is important; you have to keep it secret. You could tell only your most trusted employees the exact mixture of ingredients, but what if one of them defects to the competition? There goes the secret, and before long every grease palace on the block will be making burgers with sauce as tasteless as yours. Now think about it in another way. The recipe of burger sauce is the secret and each employee is a party. Now we can do one thing, divide the recipe into pieces for each employee, then only together can they make the burger sauce. If any employee resigns with his single piece of the recipe, his information is useless by itself. However, it has a problem, if one employee, who has a piece of the sauce recipe, goes to work for the competition and takes his piece with him, the rest of them are out of luck. He can not reproduce the recipe, but neither can work together. His piece is as critical to the recipe as every other piece combined.

(3) In a bank, there is a vault which must be opened every day. The bank employs three senior tellers; but it is not desirable to entrust the combination to any one person. Hence, we want to design a system whereby any two of the three senior tellers can gain access to the vault, but no individual can do so. This problem can be solved by means of a secret sharing scheme.

(4) Imagine you are setting up to design a control mechanism for a nuclear missile launch. There is a control panel with a key board. You can enter a secret code through the keyboard. If the secret code is correct, then the missile gets launched. There are three officers who are in charge of a missile launch. A simple solution would be to give the secret code to these three officers, but then it is possible for a lunatic officer to start a war and destroy the planet. So you are supposed to design a control mechanism with a condition that nuclear weapons could be accessed ONLY IF AT LEAST TWO of the three officers come together. So it is desirable that the secret code should be divided among the three officers in such a way that the secret code will be reconstruct only if at least two of the three officers come together. But how can you divide the secret code in such a way. This problem can be solved by means of a secret sharing scheme.

(5) Good passwords are hard to memorize. A clever user could use a secret sharing scheme to generate a set of shares for a given password and store one share in his address book, one in his bank deposit safe, leave one share with a friend, etc. If one day he forgets his password, he can reconstruct it easily. Of course, writing passwords directly into the address book would pose a security risk, as it could be stolen by an "enemy". If a secret sharing scheme is used, the attacker has to steal many shares from different places.

1.1.1 Summary of the Examples

In the above examples, we note that there is a specific party D (say) and D has a secret S (say), D wants to share the secret S among all the parties such that any set of required parties or more parties can get S by pooling their shares. But any set of less than required parties can not get any information about the secret S.

In the last two examples there is a specific party called dealer D and dealer has a secret S. Also there are three parties P_1, P_2, P_3. Dealer D wants to share the secret S among three parties such that

- Any single party cannot get S from his share.
- Any set of 2 or more parties can get S by pooling their shares.

If you can design such a method or scheme then that method or scheme is called secret sharing.

We can also think as for example one, given a secret S, we would like n parties to share the secret so that the following properties hold.

- All n parties can get together and recover S.
- Less than n parties cannot recover S.

In the map example, S is the map, while you and your friend are the two parties that share the secret.

1.2 Secret Sharing:

In cryptography, secret sharing is a method for distributing a secret among a group of participants, each of which is allocated a share of the secret. The secret can only be reconstructed when the shares are combined together; individual shares are of no use on their own. Basically our goal is to divide some secret S into n pieces $sh_1, sh_2, ...sh_n$ in such a way that knowledge of any t or more pieces makes S easily computable. Knowledge of any $t-1$ or fewer pieces leaves S completely undetermined (in the sense that all its possible values are equally likely). This scheme is called (n, t) threshold scheme. If $t = n$ then all participants are required together to reconstruct the secret.

1.3 Formal Notations for Secret Sharing:

In any secret sharing scheme, the following three entities are involved:

- A special party called dealer (D)
- A set of n players/parties $P = \{P_1, P_2...., P_n\}$
- A centralized *passive adversary* (the adversary obtains the complete information held by the corrupted players, but can not alter the information) A_t having unbounded computing power. A_t controls $t-1$ parties **(excluding D)** in *passive* fashion such that $t < n$. Secret is selected uniformly and randomly from a finite prime field F, such that $|F| > n$. All computation and communication are done over F.

NOTE: We cannot use public key cryptography, digital signatures, etc against A_t. Because A_t has unbounded computing power, and the secrecy of the public key cryptographic schemes holds good against an adversary having bounded computing power. Because the secrecy of the public key cryptographic schemes depends on the difficulty of solving certain number-theoretic hard problems. e.g. *secrecy of RSA depends on* factoring *a number, and secrecy of ELGamal depends on* Discrete logarithm. *Now if adversary has unbounded computing power then using* brute-force *algorithm adversary can solve factoring and Discrete Logarithm problems, so RSA and ELGamal breaks.*

1.4 (n, t) - Secret Sharing

Any (n, t) - secret sharing has two phases:

1 **Sharing Phase:** In this phase, D has a secret S, and D distributes the shares of S to the individual parties. See $Figure$1.1

2 **Reconstruction Phase** In this phase, parties pool their shares to reconstruct back S.

1.4.1 Properties of (n, t) - Secret Sharing:

Any (n, t) - secret sharing scheme should satisfy the following properties:

1 **Correctness:** Given any t shares, we can recover the same secret S, which was shared in the sharing phase.

2 **Secrecy:** Adversary A_t should not get any information about S during the sharing phase from the shares of $t-1$ parties under its control.

NOTE: We want to preserve the secrecy of S only up to the sharing phase. Because, S, will be any how known publicly during reconstruction phase.

Definition [6] *The view of any party P_i or adversary A_t, in secret sharing consists all the information, or we can say all the messages (private messages and broadcasts) received by him during the sharing phase. Basically view is a random variable, because every time view may not be unique. So view is a random variable follows a uniformly distribution.*

Formalization of secrecy: Suppose (A_t, S), denotes the view of the adversary A_t during sharing phase, where S is the shared secret. Any (n, t) - secret sharing protocol is secure if it satisfies the following property:

$$View\ (A_t, S_1) \equiv View\ (A_t, S_2)\ for\ any\ two\ secrets\ S_1 \neq S_2.$$

The meaning of the above property is that, the view of the adversary for two different secrets is same. That is the view for S_1 and S_2 follows identical distribution. The above property against a computationally unbounded adversary A_t is also known as *Perfect Secrecy* or *Non-Cryptographic Secrecy* or *Shannon Secrecy*. Intuitively *Perfect Secrecy* means that adversary A_t is in no better position than guessing S at the end of sharing phase.

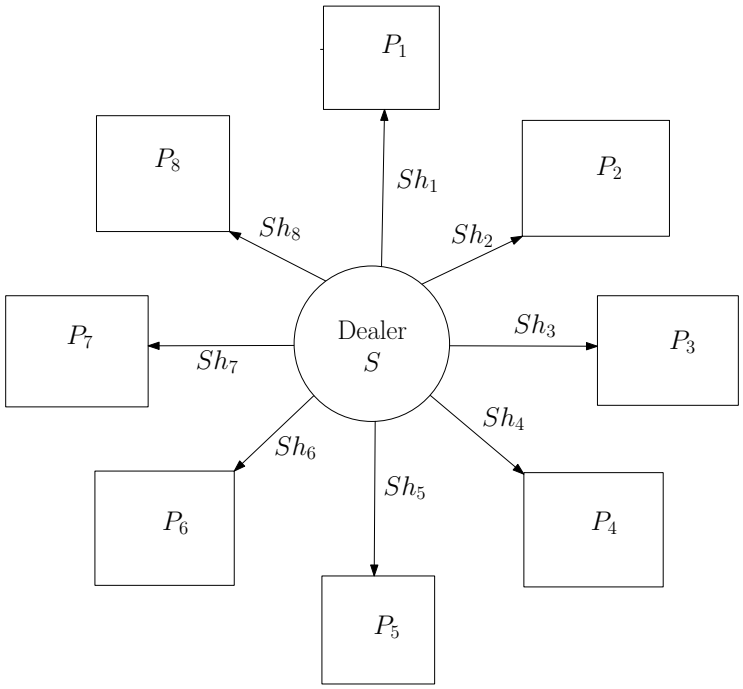

Figure 1.1: Sharing Phase of a Secret Sharing Scheme

1.5 Why do we need Secret Sharing:

The original motivation for secret sharing is the following: To **safeguard cryptographic keys** from loss, it is desirable to create backup copies. The greater the number of copies made, the greater the risk of security exposure; the smaller the number, the greater the risk that all are lost. Secret sharing schemes address this issue by allowing enhanced reliability without increased risk.

In other words we can say the motivation for secret sharing is secure key management. In some situations, there is usually one secret key that provides access to many important files. If such a key is lost (for example, the person who knows the key becomes unavailable, or the computer which stores the key is destroyed), then all the important files become inaccessible. The basic idea in secret sharing is to divide the secret key into pieces and distribute the pieces to different persons in a group, so that certain subsets of the group can get together to recover the key.

As a very simple example, consider the following scheme that includes a group of n players. Each player is given a share sh_i, which is a random bit string of a fixed specified length. The secret is the bit string

$$S = sh_1 \bigoplus sh_2 \bigoplus ..... \bigoplus sh_n$$

Note that all shares are needed to recover the secret. A general secret sharing scheme specifies the minimal sets of users who are able to recover the secret by pooling their secret information. The simple example above is a perfect n out of n secret sharing scheme.

1.6 Implementation of (n, t) - Secret Sharing

To implement (n, t) - secret sharing, one way is to use the combinatorial approach.

1.6.1 Combinatorial approach

Let $P = \{P_1, P_2, ...P_n\}$ be the set of n players. Total number of subsets with t players are $\binom{n}{t} = k$ (say). Suppose $S_1, S_2, ..., S_k$ are these subsets. Now suppose S be the secret. For each S_i, dealer sends r_j^i to the player P_j^i as a share, where $P_j^i \in S_i$ and $S = r_1^i + r_2^i + ... + r_t^i$. Do the same for each $i = 1, 2, ..., k$. For reconstruction any t players come and pool their shares. But combinatorial approach for implementing (n, t)- secret sharing is impractical and infeasible to implement. A popular technique to implement (n, t) secret sharing uses polynomial interpolation ("Lagrange interpolation"). This method was invented by Adi Shamir in 1979.

1.6.2 (n, t) - Shamir Secret Sharing

The essential idea of Adi Shamir's threshold scheme [8] is that 2 points are sufficient to define a line, 3 points are sufficient to define a parabola or a circle, 4 points to define a cubic curve and so forth. That is, it takes t points to define a polynomial of degree $t-1$. Shamir's secret sharing scheme is a threshold scheme based on polynomial interpolation.

Let us suppose that there is a dealer D with secret S. Dealer D wants to share the secret S among the n players $P_1, P_2, ..., P_n$ such that any $t \leq n$ players are required to reconstruct the secret S. But no group of $t-1$ participants can do so.

1 **Sharing Phase:** D chooses a random polynomial $f(x)$ with degree $t-1$ over the finite field F with secret $S = f(0)$. We require that $|F| = p \geq n+1$, where p is a prime or prime power. D *publicly* choose n random non-zero distinct values x_i from the field F, and secretly distributes to each player P_i the share $sh_i = f(x_i)$.

2 **Reconstruction Phase:** Any t players pool their shares. Without loss of generality assume that first t players pool their shares, $sh_1, sh_2, ..., sh_t$. Now use Lagrange interpolation to find the unique polynomial $f(x)$ such

that $degree f(x) < t$ and $f(x_i) = sh_i$ for $i = 1, 2, ..., t$. Reconstruct the secret to be $f(0)$.

Lemma 1.6.1 (Lagrange interpolation formula for polynomials) *Suppose given a set of t data points:*
$(x_1, y_1), (x_2, y_2), ...(x_i, y_i), ...(x_t, y_t)$
where no two x_i are the same, the interpolation polynomial in the Lagrange form is given as

$$f(x) = \sum_{i=1}^{t} y_i L_i(x)$$

Where $L_i(x)$ is given as

$$L_i(x) = \frac{(x-x_1)(x-x_2).....(x-x_{i-1})(x-x_{i+1}).....(x-x_t)}{(x_i-x_1)(x_i-x_2).....(x_i-x_{i-1})(x_i-x_{i+1}).....(x_i-x_t)}.\ \forall i = 1, 2, ...t$$

NOTE: how, given the initial assumption that no two x_i are the same, so this expression is always well-defined.

Lemma 1.6.2 (Uniqueness) *There is a* unique *polynomial which satisfies the points:*
$(x_1, y_1), (x_2, y_2), ...(x_i, y_i), ...(x_t, y_t)$

Proof Suppose $f(x) = a_0 + a_1 x + a_2 x^2 + ... + a_{t-1} x^{t-1}$ .....(1)
be the polynomial which satisfies the above points.
$f(x)$ interpolates the data points it's mean that $f(x_i) = y_i$, $\forall i = 1, 2, ...t$
If we substitute equation (1) in here, we get a system of linear equations in the coefficients of $f(x)$.

The system in matrix-vector form can be written as

$$\begin{bmatrix} x_1^{t-1} & x_1^{t-2} & x_1^{t-3} & ..... & x_1 & 1 \\ x_2^{t-1} & x_2^{t-2} & x_2^{t-3} & ..... & x_2 & 1 \\ .. & .. & .. & ..... & .. & .. \\ .. & .. & .. & ..... & .. & .. \\ .. & .. & .. & ..... & .. & .. \\ x_t^{t-1} & x_t^{t-2} & x_t^{t-3} & ..... & x_t & 1 \end{bmatrix} \begin{bmatrix} a_{t-1} \\ a_{t-2} \\ .. \\ .. \\ .. \\ a_0 \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \\ .. \\ .. \\ .. \\ y_t \end{bmatrix}$$

We have to solve this system for a_i to construct the interpolant $f(x)$. The matrix on the left is commonly referred to as a **Vandermonde matrix**. Without loss of generality suppose the above system is $\mathbf{Ax=b}$.
where A is Vandermonde matrix, now determinant of A is given as:

$$det(A) = \prod_{i,j=1, i<j}^{t} (x_i - x_j)$$

since the t points are distinct, the determinant can't be zero as $x_i - x_j$ is never

zero, therefore A is nonsingular therefor the system has a unique solution.

Theorem 1.6.3 *Shamir's secret sharing scheme satisfies both the property of* (n, t) *secret sharing, namely* correctness *and* secrecy*.*

Proof Now we will proof the properties.

1 **Correctness:** Correctness is straightforward. We use Lemma 1.6.1 to find the secret, we always get unique polynomial by Lemma (1.6.2).

2 **Secrecy:** Now we need to verify the *Secrecy* of this scheme. Suppose we have only $t-1$ parties contributing shares. This corresponds to knowing $t-1$ points of a degree $t-1$ polynomial. Can we find out coefficient $f(0) = S$? Or even gain partial information about secret? It turns out we cannot. Stating this formally, given $t-1$ shares $(x_i, f(x_i))$, and a hypothetical value x^* for the secret, to test whether the secret is x^*. We know that secret is constant term of $f(x)$, so we need that $x^* = f(0)$, or in other words, that point $(0, x^*)$ is another correspondence. If we just know $t-1$ points, none of which have $x_i = 0$.Thus all x^* values from the field F, for the secret are equally likely, and secrecy holds.

Note that Shamir's scheme is *provable secure*, that means: in a (n, t) scheme one can prove that it makes no difference whether an attacker has $t-1$ valid shares at his disposal or none at all; as long as he has less than t shares, there is no better option than guessing to find out the secret.

1.6.3 Graphical View:

Now again we will briefly explain Shamir's secret sharing in graphical view, as it will help to clear the concept.

Sharing Phase: In the sharing phase dealer chooses a random polynomial $f(x)$ of degree $t-1$, and take n points on this polynomial. These polynomial points dealer sends to the players. See *Figure* 1.2

Reconstruction Phase: In the reconstruction phase any t players come together, without the loss of generality assume first t players pool their shares, that is the points on the polynomial. Now we have t points, we can reconstruct the polynomial $f(x)$. See *Figure* 1.3

1.6.4 Observations:

1. In shamir's secret sharing why we choose prime?
   – Because the cardinality of finite field is prime or prime power.

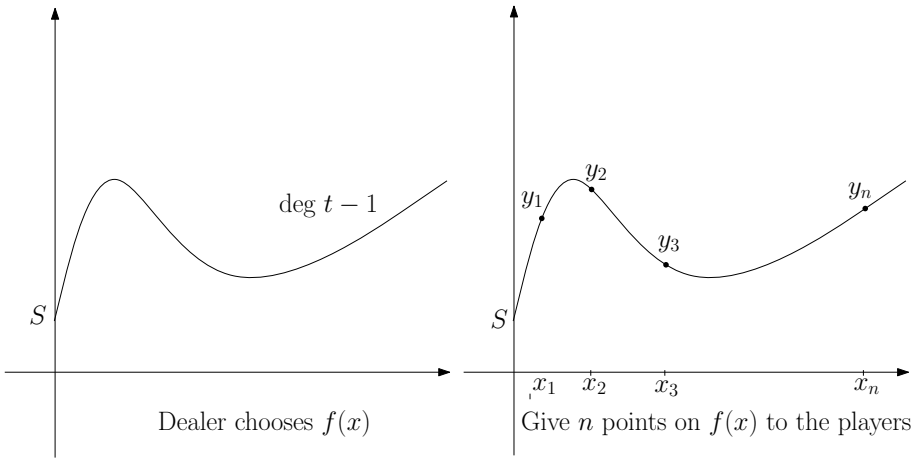

Figure 1.2: Sharing Phase

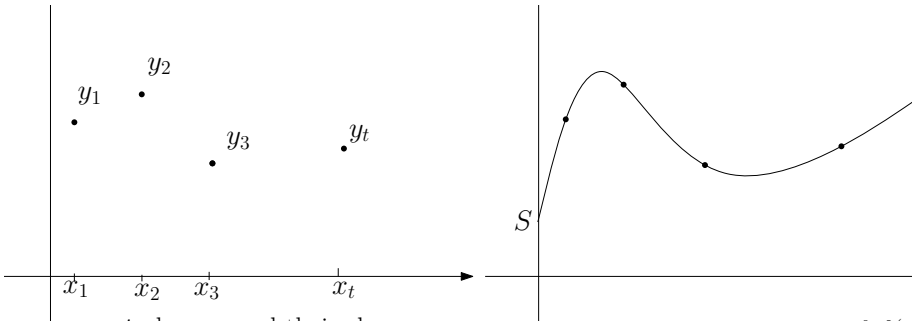

Figure 1.3: Reconstruction Phase

2. Why we choose field, why not any other algebraic structure?
   – Because field gives us facility to do operations namely, addition, multiplication, substraction, divide. Other algebraic structure does not provide these facilities.

3. Why we choose prime $p \geq n + 1$?
   – We need to choose n non zero distinct points from the field. As field consists of zero element, If cardinality of field is less than $n + 1$, we can not choose n non zero distinct points from the field.

4. Why we choose non-zero element?
   Otherwise player with 0 element can get the complete secret as: $f(0) = S$.

5. Why we choose distinct element?
   – Otherwise more than one player will get same share, this situation

can create problem in reconstruction of the polynomial.

1.6.5 Examples:

Example (1) -*Sharing Phase:* Suppose $S = 1234$ be the secret. Number of players $n = 6$ and threshold $t = 3$. Now suppose dealer chooses a polynomial
$f(x) = 1234 + 166x + 94x^2$
over a sufficient large field. Note secret $S = f(0) = 1234$.
For simplicity assume that dealer chooses publicly non zero distinct elements $1, 2, 3, 4, 5$. Therefore Secret share points are:
$(1, 1494), (2, 1942), (3, 2598), (4, 3402), (5, 4414), (6, 5614)$.
Dealer gives each participant a different single point (both x and $f(x)$).

Reconstruction Phase: In order to reconstruct the secret any 3 points will be enough. Let us consider
$(x_1, y_1) = (2, 1942), (x_2, y_2) = (4, 3402), (x_3, y_3) = (5, 4414)$.
Using Langrange polynomial:
$L_1(x) = \frac{(x-x_2)(x-x_3)}{(x_1-x_2)(x_1-x_3)} = \frac{1}{6}x^2 - \frac{11}{2}x + \frac{31}{3}$
$L_2(x) = \frac{(x-x_1)(x-x_3)}{(x_2-x_1)(x_2-x_3)} = -\frac{1}{2}x^2 - \frac{31}{2}x - 5$
$L_3(x) = \frac{(x-x_1)(x-x_2)}{(x_3-x_1)(x_3-x_2)} = \frac{1}{3}x^2 - 2x + \frac{22}{3}$

$f(x) = y_1L_1(x) + y_2L_2(x) + y_3L_3(x) = 1234 + 166x + 94x^2$
So secret S is $f(0) = 1234$.

Example (2) The bank vault example and nuclear missile launch example can be solved using $(3, 2)$ - Shamir secret sharing. As we know in these examples there are three players, and dealer wants to share secret S.
Dealer chooses randomly a straight line, and gives three points to the play-

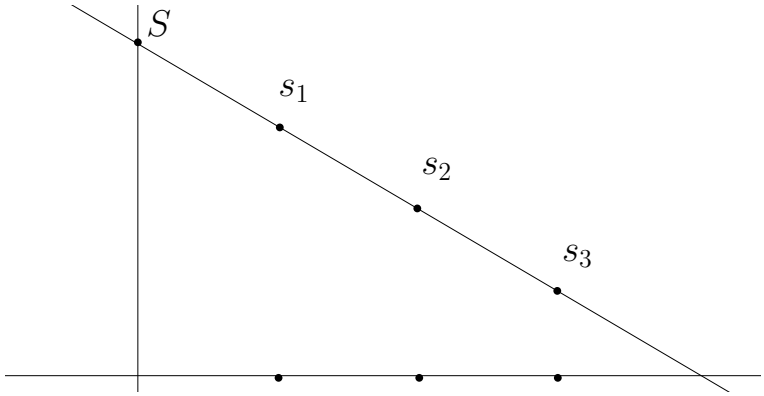

Figure 1.4: Sharing Phase

ers on this line. If at least two parties pool their shares then the straight line can be uniquely interpolated. But with only one share, say (α_1, s_1), there are all possible straight lines in the field F passing through this point. See *Figure* 1.5

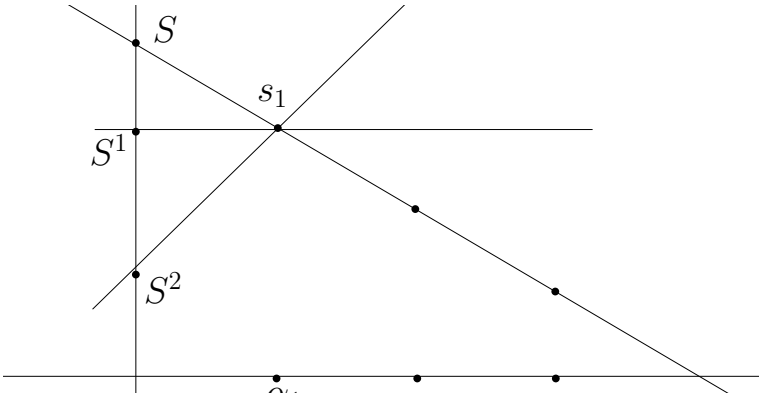

Figure 1.5: Secrecy preserved

1.6.6 Limitations of Shamir Secret Sharing:

It seems we have a solution for secret sharing, we have an efficient procedure to share a secret such that secrecy and correctness both hold. Looks like we're done, right? Actually, we should not be satisfied with Shamir's scheme. Here are some limitations with it.

- In Shamir secret sharing, D is assumed to be honest.
- Also adversary A_t is assumed to be passive.
- If the participants cheat in the recover phase, the secret can not be recovered. The other participants don't even have a way of knowing if someone cheated.

In real life scenario, A_t can be active.

Definition *(Active Corruption)* A_t has full control over the corrupted parties and can force the corrupted parties to behave in any arbitrary manner during a protocol.

1.6.7 Problems created by active adversary:

Sharing Phase: In the sharing phase of Shamir secret sharing dealer distributes the shares. We assumed that dealer D is honest, and honestly chooses

the polynomial $f(x)$ of $degree(t-1)$, but D may be corrupted and can choose polynomial $f(x)$ of $degree \geq t$, then how can t players reconstruct the same polynomial $f(x)$. Since t points can reconstruct a polynomial of degree at most $t-1$. In other words, the dealer may not a share valid secret and may get away with it.

Reconstruction Phase: Suppose D is honest but one or more parties are dishonest. That is some players are actively corrupted. In reconstruction phase actively corrupted players may produce wrong shares.

The above two problems clearly say that secret sharing is not equipped to tolerate these problems.

Chapter 2

Verifiable Secret Sharing

In chapter 1, we stated, if dealer or participants are cheater then there arise two problems:

1. In the sharing phase, the dealer might send inconsistent shares to the players.
2. In the reconstruction phase players can give wrong shares.

In other words we can say that if adversary is *actively* corrupted (adversary takes full control over the corrupted players) then there is problem to reconstruct the secret. To overcome these problems, the first effort came from *Tompa and Woll [3] and McEliece and Sarwate* [4], who gave some partial solutions. After that, the notion of VSS was introduced by *Chor, Goldwasser, Micali and Awerbuch* [2] to completely resolve the problems.

2.1 Dealing with active adversary

We should incorporate *verification mechanism* which ensures the following:

1. D has selected a t degree polynomial and distributed shares on this polynomial during the sharing phase.
2. The share produced by a party during the reconstruction phase is same as the one received during the sharing phase. It means, it should be verified that parties produce right shares.

Secret sharing (SS) + *above verification mechanism* = *Verifiable Secret Sharing* (VSS)

2.2 Verifiable Secret Sharing

Let $P = \{P_1, P_2, ...P_n\}$ be the set on n players, and D be the dealer. Now there are two models:

1. $D \in P$
2. $D \notin P$.

We will assume (1) $D \in P$, also assume that at most t players can actively corrupted (possibly including D). Now we define the verifiable secret sharing. VSS is a two phase protocol (Sharing and Reconstruction) carried out among n parties in the presence of a active adversary. The goal of the VSS is to share a secret, S, among the n parties during the sharing phase in a way that would later allow for a unique reconstruction of this secret in the reconstruction phase, while preserving the secrecy of S until the reconstruction phase.

Sharing Phase: The dealer initially holds secret $S \in F$ where F is a finite field of sufficient size; and each player P_i holds some private information r_i. The sharing phase may consist of several rounds. In each round, each player can privately send messages to other players or can broadcast a massage.

Reconstruction Phase: In the reconstruction phase, each player P_i reveals his share r_i^* (a dishonest player may reveal $r_i^* \neq r_i$), and a reconstruction function Rec is applied in order to compute the secret, $S = Rec(v_1^*, ..., v_n^*)$.

2.2.1 VSS Properties:

Any VSS protocol should satisfy the following properties:

Secrecy: If D is *honest* (i.e., is uncorrupted by the adversary), then the adversary's view during the sharing phase reveals no information about S. More formally, the adversary's view is identically distributed under all different values of S. It means, if D is honest, then secret S is perfectly secure during the Sharing phase. Note we need secrecy if D is honest and up to the sharing phase.

Correctness: If D is *honest*, then the reconstructed value is always equal to the secret S, irrespective of the behavior of adversary.

Strong commitment: If D is *dishonest*, then there exits some $S^* \in F$ such that S^* is committed/shared by D during sharing phase, and same S^* will be reconstructed during reconstruction phase, irrespective of the behavior of adversary.

Now there arise some questions,

1. How the corruption is done?
   –It depends on the *adversary model*.

2. How we can verify that secret distribution is consistent?
   –To verify this there is need for communication between players, so *communication model* is required.

2.3 Communication Model

Communication model [6] has some attributes as *Medium of Communication, Network Topology, Control over Channels, Synchrony of Network* etc.

2.3.1 Medium of Communication

In any VSS protocol, the parties communicate with each other over channels where channels can be *point-to-point, multi-cast and broadcast.*

Point-to-point channels: It is a *one to one* channel, which enables both way communication between two players.

Multi-cast channels: It is a *one to many* channel, which allows a player to send some message identically to a subset of players in the network.

Broadcast channels: It is a *one to all* channel, which allows any player to send some message identically to all other players in the network.

2.3.2 Network Topology

The topology of the network means, how players are connected to each other. Basically a network can be of two types *complete or incomplete.*

Complete network: In a complete network, every pair of players are directly connected. That is all players form a complete graph.

Incomplete network: In an incomplete network, the connectivity can be limited. That is every pair of players may not be directly connected.

2.3.3 Control over Channels

Communication channels can be categorized according to the control of the adversary over channels. We distinguish three levels of control over the channel, *Secure, Insecure, and Unauthenticated.*

Secure: Channel is secure, it means, channel is authentic and secret. In simple words we can say that in secure channel model, the communication between any two uncorrupted or honest players are completely out of reach to the adversary i.e adversary cannot affect or change or even eavesdrop the communication.

Insecure: Channel is insecure, it means, channel is authentic but tappable. In simple words we can say that in insecure channel model, the adversary can hear all the communication among all the parties; yet the adversary can not alter the communication between two honest parties.

Unauthenticated: Channel is unauthenticated, it means, channel is unauthenticate and tappable. In simple words we can say that in unauthenticated channel model, the adversary has full control over the communication. That is, on the top of tapping the communication the adversary can delete, generate and modify messages at wish.

2.3.4 Synchrony of the network

There are two types of network setting *Synchronous setting and Asynchronous Setting*.

Synchronous setting: In synchronous setting there is concept of global clock. It means if a player sends a massage to another player, then time delay is bounded by some known constant.

Asynchronous setting: In asynchronous setting there is no concept of global clock. It means if a player sends a massage to another player, then there is no fix time for massage delay. Moreover, messages may be received in an order different than the order of sending.

2.4 Adversary Model

In adversary model [6], we need to know, what is the *computational power* of adversary, *control over the corrupted players* by adversary, *mobility of adversary*, and *corruption capacity* of the adversary. We will discuss all these one by one.

2.4.1 Computational power

The computational power of the adversary may be limited to *probabilistic polynomial time* or may have *unbounded computing power*. Adversary with polynomial time power is called *bounded adversary*, and with unbounded computing power is called *unbounded adversary*.

2.4.2 Control over the corrupted players

Control over the corrupted players [7] means, what can the adversary do with the information of the corrupted players. An adversary can control over the players in four ways. According to these four ways, the adversary can be of four type, namely *passive, active, fail-stop and mixed adversary*

Passive adversary: If the adversary acts like an eavesdropper, that is adversary can try to know the data of honest players on the basis of information

with corrupted players, but can not alter the data of corrupted players. Such an adversary is called as *passive* adversary.

Active adversary: If the adversary can take complete control of the corrupted players and can alter the behavior of the corrupted players in an arbitrary fashion. Such an adversary is called *active* adversary.

Fail-stop adversary: If the adversary can stop the working of any of the corrupted player. Such an adversary is called as *fail-stop* adversary.

Mixed adversary: If the adversary may simultaneously control some players in passive, active and fail-stop fashion (possibly disjoint set of players). Such an adversary is called mixed adversary.

2.4.3 Mobility of adversary

Mobility of adversary means, at which stage, during the protocol adversary can corrupt the players, according to that stage we can categorize the adversary. According to mobility of adversary, there can be three types of adversary, namely *static, adaptive and mobile.*

Static adversary: If the set of corrupted players is fixed by adversary, before the protocol begins its execution (but set of corrupted players unknown to us), then such an adversary is called as a static adversary. Once a party is corrupted by static adversary, he remains corrupted for the rest of the protocol execution.

Adaptive adversary: If the adversary is allowed to corrupt players during the protocol execution, then such an adversary is called as an adaptive adversary. Thus an adaptive adversary chooses which player to corrupt as the protocol proceeds. Once a party is corrupted, he remains corrupted for the rest of the protocol execution.

Mobile adversary: If an adversary can corrupt players at any time, but he can also release corrupted players, regaining the capability to corrupt further players, such an adversary is called as a mobile adversary. Thus an adversary is mobile if he can corrupt, in an adaptive way, a different set of parties at different times during the execution. That is a party once corrupted need not remain so throughout.

2.4.4 Corruption capacity of the adversary

Corruption capacity of an adversary is defined as the maximum number of players, which can be corrupted by adversary. There are two different ways of specifying the number of corrupted players, namely *threshold and non-threshold.*

Threshold: In the threshold specialization, the number of corrupted players, at any given time, is limited to at most t (a threshold).

Non-threshold The non-threshold specialization is a generalization of the threshold one. In the non-threshold specialization, an adversary structure which is a set of subsets of the players, is used where the adversary is permitted to corrupt the players of any one arbitrarily chosen subset in the adversary structure.

2.5 Types of Security

According to the computing power of adversary, we can categorize the security into two types. Security against the adversary having unlimited computing power, is called *information-theoretic security.* While security against an adversary having limited computing power, is called *cryptographic security.*

2.6 Environment for VSS Protocols use in this Thesis

In this thesis, we will survey VSS protocols in the following setting:

- **Communication Model**
  1. **Medium of Communication:** Point-to-point channel with and sometime without broadcast channel.
  2. **Network Topology:** Complete network
  3. **Control over Channels:** Secure channel model.
  4. **Synchrony of Network:** synchronous
- **Adversary Model**
  1. **Computational power:** Unbounded powerful adversary.
  2. **Control over the Corrupted Players:** Active
  3. **Mobility:** Static
  4. **Corruption Capacity:** Threshold

NOTE: We will talk about information-theoretic security, because we are assuming adversary with unbounded computing power. We denote A_t as the adversary with the above features, where t is the threshold for corruption.

Still before survey of VSS protocols we need to know some thing more. Any VSS protocol under the above setting has four parameter namely, *Resilience, Communication Complexity, Round Complexity* and *Computation Complexity.*

- **Resilience:** Resilience is the maximum number of corrupted players that can be tolerated by the protocol, and still protocol satisfy its all three properties *secrecy, correctness* and *strong commitment.*

- **Communication complexity:** Communication complexity tries to quantify the amount of communication required for the protocol. It is the total number of bits communicated by the honest parties in the protocol. A protocol is called communication efficient if the communication complexity is polynomial in n and error parameter (in case the protocol is statistical and has an error parameter).

- **Round complexity:** Round complexity is the total number of rounds taken for the execution of the protocol in sharing phase. Generally all the VSS protocol take only one round to reconstruct the secret. The round complexity of VSS protocols is one of their most important complexity measures. Indeed, interaction over a computer network is usually the most time-consuming operation (because of lagging or network congestion). It is thus very important to devise protocols which require the minimal number of rounds to complete. A protocol is called round efficient if the round complexity is polynomial in n and the error parameter (in case the protocol is statistical and has an error parameter).

- **Computation complexity:** It is the computational resources required by the honest parties during a protocol execution. A protocol is called computationally efficient if the computational resources required by each honest party are polynomial in n and error parameter (in case the protocol is statistical and has an error parameter).

2.7 Types of VSS

In information theoretic settings (i.e. under the assumption of a computationally unbounded adversary), there are mainly two flavors of VSS namely *Perfect* VSS (i.e. error free) and *statistical* VSS (involves some probability of error)

Perfect VSS: A VSS protocol is said to be perfect VSS, if the protocol satisfies all the three properties *secrecy*, *correctness* and *strong commitment*, without any error probability.

Statistical VSS: A VSS protocol is said to be statistical VSS, if the VSS protocol satisfies the *secrecy*. But protocol satisfies two other properties *correctness* and *strong commitment* with some error probability.
NOTE: We assume secrecy to be perfect

Theorem 2.7.1 *[14] Perfect VSS protocol, tolerating adversary A_t is possible if and only if $n \geq 3t + 1$.*

Theorem 2.7.2 *[13] Statistical VSS protocol, tolerating adversary A_t is possible if and only if $n \geq 2t + 1$.*

2.8 Conclusion

In this chapter we looked into the definition of the VSS. Also we discussed the different settings for the VSS protocols. Finally in the section 2.6 we mentioned the environment of our interest, for the VSS protocols. In the upcoming chapters we will study the perfect VSS protocols in this environment.

Chapter 3

Three-Round VSS Protocol with $n \geq 4t + 1$

In this chapter we will discuss the three-round VSS protocol [5] with $n \geq 4t+1$ under the setting, which we considered in chapter 2. The protocol is designed using bivariate polynomials. So first we need to know the properties of the bivariate polynomial.

3.1 Properties of Bivariate Polynomials

Let us suppose that $F(x, y)$ be the random bivariate polynomial over the field F of degree t in both x and y. Suppose $F(x, y)$ is given as:

$$F(x,y) = \sum_{i=0,j=0}^{i=t,j=t} r_{ij}x^i y^j \text{ , where } r_{ij} \in F$$

- $F(x, y)$ can have at most $(t + 1)^2$ coefficients.
- If we know $(t + 1)^2$ points on the bivariate polynomial $F(x, y)$, but we don't know, what is $F(x, y)$, then we can reconstruct uniquely $F(x, y)$.
- Let $f_i(x) = F(x, i)$ and $g_i(y) = F(i, y)$, where $i, j \in F$, then both $f_i(x)$ and $g_i(y)$ are univariate polynomial of degree t.
- $f_i(j) = g_j(i)$

NOTE: Each $f_i(x)$ is the i^{th} row polynomial, and each $g_j(y)$ is the j^{th} column polynomial of the Table 3.1*, $\forall i, j \in \{1, 2, ..., n\}$. It simply means that, the values in the i^{th} row are the values on the polynomial $f_i(x)$, and similarly values in the j^{th} column are the values on the polynomial $g_j(y)$. We have $n \geq t$ values*

Table 3.1: n^2 points on $F(x, y)$

	$g_1(y)\downarrow$	....	$g_j(y)\downarrow$	....	$g_n(y)\downarrow$
$f_1(x)\rightarrow$	$F(1,1)$	....	$F(j,1)$	....	$F(n,1)$
....	....	....	....	....	....
$f_i(x\rightarrow$	$F(1,i)$	....	$F(j,i)$	....	$F(n,i)$
....	....	....	....	....	....
$f_n(x)\rightarrow$	$F(1,n)$	....	$F(j,n)$	....	$F(n,n)$

in each row and in each column, and degree of $f_i(x)$ and $g_j(y)$ is t. So we can easily reconstruct both $f_i(x)$ and $g_j(y)$.

3.2 Protocol

Sharing phase: Three Rounds

- *Round* (1)
  1. Dealer D chooses a random bivariate polynomial $F(x, y)$ over the field F, with degree t in x and y such that $F(0, 0) = S$, where $S \in F$ is the secret.
  2. Dealer D *privately* sends to every player P_i the univariate polynomials $f_i(x) = F(x, i)$ and $g_i(y) = F(i, y)$.
- *Round* (2)
  Every player P_i *privately* send to P_j the value $g_i(j)\ \forall j \in \{1, 2, ...n\}$.
- *Round* (3)
  Player P_j *broadcast* a list L_j of players P_i for whom it holds that $f_j(i) \neq g_i(j)$.

Local computation by each player:

1. Every player construct a graph G on n nodes. In G, the edge (i, j) exists if P_i is not in the list L_j and P_j is not in the list L_i.
2. Find the complementary graph of G.
3. Find the maximal matching in the complementary graph
4. Define C to be G without the vertices of the matching.
5. Define ADD to be the set of vertices i such that $i \neq C$ and there exits $2t + 1$ nodes $j \in C$ such that $i, j \in G$.
6. If $|C| + |ADD| \geq 3t + 1$ the accept the sharing, otherwise disqualify the dealer.

Reconstruction Phase:
Each player $P_i \in C \cup ADD$ *broadcasts* $f_i(0)$. Use error correction on $\{f_i(0)\}_{i \in C \cup ADD}$ to recover the polynomial $g_0(y) = F(0, y)$. The reconstructed value is $g_0(0)$.

3.3 Analysis of the Protocol

Sharing Phase:

Round (1)**:** In the first round dealer simply chooses a random polynomial $F(x, y)$, and hide the secret S as the constant term of the polynomial $F(x, y)$. Because dealer chooses $F(x, y)$ randomly, so players do not know the $F(x, y)$. Now dealer shares the secret S by sending univariate polynomials to the players. Indices of the players are fixed and publicly known.

Round (2)**:** Now we need to check whether the shares distributed by dealer in round (1) are consistent or not. Because dealer may also be corrupted. For this there is need for communication between the players. So every player P_i privately sends to P_j the value $g_i(j)$, because P_i will be consistent with P_j iff $g_i(j) = f_j(i)$ and $g_j(i) = f_i(j)$.

Round (3)**:** Every player P_i construct a list L_i of inconsistent players, and broadcast this list.
NOTE: Still we can not say any thing about L_i*, even we can not say any thing about cardinality of* L_i*, because a dishonest player can broadcast any list with cardinality* 0 *to* n

Local computation by each player: In third round of sharing phase every player broadcast the list L_i. So every player has a set of lists, $L = \{L_1, L_2, ...L_n\}$. Now draw a graph G with n vertices and there is an edge (i,j) iff P_i is not in the list L_j, and P_j is not in the list L_i.
NOTE: If dealer has shared something consistent then in graph G *there will be a clique of size at list* $3t + 1$
But to find the clique in a graph is a NP-hard problem, so we use a trick here, for that we need to know some basic definitions from graph theory.

- **Definition** *(Compliment Graph): In graph theory, the complement or inverse of a graph* G *is a graph* H *on the same vertices such that two vertices of* H *are adjacent iff they are not adjacent in* G*.*
- **Definition** *(Maximal Matching): Given a graph* $G = (V, E)$*, a matching* M *in* G *is a set of pairwise non-adjacent edges; that is, no two edges share a common vertex. A maximal matching is a matching* M *of a graph* G *with the property that if any edge not in* M *is added to* M*, it is no longer a matching, that is,* M *is maximal if it is not a proper subset of any other matching in graph* G*.*

- **Definition** *(Clique): A clique in an undirected graph G is a subset C of the vertex set of G, such that for every two vertices in C, there exists an edge connecting the two.*

Reconstruction Phase: To reconstruct the share S, each player $P_i \in C \cup ADD$ *broadcasts* $f_i(0)$. Now among these $f_i(0)$ values at most t may be corrupted, and $|C \cup ADD| \geq 3t+1$, also note that $f_i(0) = g_0(y) = F(0, y)$. By using error correcting algorithm we can construct $g_0(y)$, and secret is $S = g_0(0)$.

Lemma 3.3.1 *If dealer is honest then C includes at least $2t+1$ honest players.*

Proof If D honest then there is a clique between all honest players. This means that each edge in the matching includes at least one dishonest player. Thus there can be at most t honest players in the matching, and all are the rest are in C. Thus, there are at least $2t + 1$ honest players in C.

Lemma 3.3.2 *If dealer is honest then each honest player P_i not in C will be included in ADD.*

Proof If D is honest then by above lemma 3.3.1, there are at least $2t+1$ honest players in C. Also we know that, in the honest dealer case, all honest players are a clique. If P_i is honest and $P_i \notin C$, then P_i will be connected to at least $2t + 1$ players in C and hence P_i will be add to ADD.

Lemma 3.3.3 *Even if dealer is dishonest and dealer does not get disqualified in the sharing phase, then also the set of honest players in $C \cup ADD$ defines a unique bivariate polynomial $F^*(x, y)$ of degree at most t in x and y.*

Proof Let us suppose that D does not disqualified in the sharing phase, then $|C| \geq 2t + 1$. Otherwise $|C| + |ADD| < 3t + 1$ and D should not have qualified. Now C contains at least $t + 1$ honest players. Suppose these honest players defines a polynomial $F^*(x, y)$. According to definition of ADD, every other honest player $P_i \in ADD$ agree with at least $2t + 1$ values in C, out of which at least $t+1$ belong to honest players. Thus it is on $F^*(x, y)$. Take $S^* = F^*(0, 0)$. We need to prove that this value will be reconstructed by all the good players. We know that $|C| + |ADD| \geq 3t + 1$ and all good players in $C \cup ADD$ defines this secret S^* and there are at most t bad players in $C \cup ADD$. Thus, the error correcting procedure will yield the values S^*.

Theorem 3.3.4 *The three round protocol satisfies all the three properties of VSS.*

Proof We will show that all the three properties of the VSS are satisfied by the protocol. As we know that the *secrecy* and the *correctness* properties make sense, if adversary is *honest*. Third property *strong commitment* is required to be shown when dealer is *dishonest*. so we will take two cases.

- *HONEST DEALER:*

  Secrecy: We need to show that the secrecy is preserved up to sharing phase. For this we analysis each round of sharing phase. In *Round (1)* secrecy preserve, because even if adversary controls over t players, then adversary has knowledge of t, $f_i(x)$ and $g_i(y)$ pairs. But to reconstruct the secret S, which is the constant term of $F(x, y)$, adversary needs at least such $t + 1$ pairs. So secrecy is preserved. Adversary can only guess the value of the secret, and that guess can be any value from the field F. In *Round (2)* player P_i sends to player P_j the value $g_i(j)$, then also adversary can not get any information about the secret. Because even if P_j is corrupted then also he is receiving $g_i(j) = f_j(i)$ value from P_i that he has already obtained. In *Round (3)* also no information is revealed related to the secret. Now only we need to show that dealer does not get disqualified. From *Lemma* 3.3.1 and *Lemma* 3.3.2 we note that $|C| + |ADD| \geq 3t + 1$ will hold if D is honest. So dealer will not be disqualified.

  Correctness: It is straight forward. If dealer is honest then from *Lemma* 3.3.1 and *Lemma* 3.3.2 we know that $|C| + |ADD| \geq 3t + 1$ and all honest players are in one of the sets. In the reconstruction phase at least $2t + 1$ of the $3t + 1$ values are the true values of $g_0(y)$ hence the *error correction recovers this polynomial and the secret S*

- *DISHONEST DEALER:*

  Strong commitment: If D is dishonest then there are two cases, either D disqualified or qualified. If D disqualified then protocol has terminated properly. So we think only when D was not disqualified. That is $|C| + |ADD| \geq 3t + 1$. Now we need to show that there exist a consistent secret S^* and all good players reconstruct S^*. It is proved by *Lemma* 3.3.3. Hence proof of the theorem completes.

3.4 Reducing the no. of rounds

In this section we will study, how can we reduce the number of rounds from three to two. To reduce the number of rounds Gennaro et.al [5] introduced a technique. According to this technique in round one when dealer sends shares, in the same time players exchange random pads via private channels. Now in the next round instead of sending values via private channels, players can broadcast their values by using random pad. Now at this point we can easily check where is inconsistency occurred.

3.4.1 Two-Round Protocol with $n \geq 4t + 1$

Sharing phase; Two Rounds:

- *Round* (1)

1. Dealer D chooses a random bivariate polynomial $F(x, y)$ over the field F, with degree t in x and y such that $F(0, 0) = S$. where $S \in F$ be the secret.
2. Dealer D *privately* sends to every player P_i the univariate polynomials $f_i(x) = F(x, i)$ and $g_i(y) = F(i, y)$.
3. Player P_i sends to each player P_j, an independent random pad r_{ij} picked uniformly from the field F.

- *Round* (2)
  Every player P_i *broadcasts*
  - $a_{ij} = f_i(j) + r_{ij}$
  - $b_{ij} = g_i(j) + r_{ji}$

Local computation by each player:

1. Define L_i to include all players P_j for whom $b_{ij} \neq a_{ji}$.
2. Every player construct a graph G on n nodes, in G, exits an edge (i, j) if P_i is not in the list L_j and P_j is not in the list L_i.
3. Find the complementary graph of G.
4. Find the maximal matching in the complementary graph
5. Define C to be G without the vertices of the matching.
6. Define ADD to be the set of vertices i such that $i \neq C$ and there exits $2t + 1$ nodes $j \in C$ such that $i, j \in G$.
7. If $|C| + |ADD| \geq 3t + 1$ the accept the sharing, otherwise disqualify the dealer.

Reconstruction Phase:
Each player $P_i \in C \cup ADD$ *broadcasts* $f_i(0)$. Use error correction on $\{f_i(0)\}_{i \in C \cup ADD}$ to recover the polynomial $g_0(y) = F(0, y)$. The reconstructed value is $g_0(0)$.

3.5 Conclusion

In this chapter we discussed a 3-round VSS protocol with $n \geq 4t + 1$ given by Gennaro et. al [5]. Also we studied a round reduction technique, and discussed a 2-round VSS protocol with $n \geq 4t + 1$. In the next chapter, we will discuss an inefficient 3-round protocol with $n \geq 3t + 1$

Chapter 4

Inefficient 3-Round VSS Protocol with $n \geq 3t + 1$

In this chapter we will discuss an inefficient three round protocol of [5] with $n \geq 3t + 1$. Suppose $P = \{P_1, P_2, ...P_n\}$ be the set of n players and t be the threshold value. Now first we will look at two results and then we will start protocol.

Theorem 4.0.1 *A perfect VSS protocol can be designed iff $n \geq 3t+1$. That is no matter whatever be the size of the network but this network can be corrupted up to 33%. So we can design perfect VSS protocol for $n \geq 3t+1, n \geq 4t+1, n \geq 5t+1$ etc. But we can not design perfect VSS protocol for any $n < 3t + 1$.*

Theorem 4.0.2 *The minimum number of rounds needed to design a perfect VSS protocol for $n = 3t + 1$ is 3, i.e.if network is corrupted up to33%, then we need at least 3 rounds to design a VSS protocol.*

4.1 Idea of the Protocol:

Very first question comes in our mind, is it possible to design such a protocol? Our answer will be positive, yes by *Theorem* (4.0.1) we can say , it is possible. Now we think a little more, is it possible to design two-round perfect VSS protocol for $n = 3t + 1$? This time our answer will be negative , no we can not design because of *Theorem* (4.0.2). Ok one more question, can we design a more than three-round perfect VSS protocol for $n \geq 3t + 1$? Of course yes, because three-round is the lower bound.
Now we start the idea of the protocol, let us suppose that $P_1, P_2, ...P_n$ be the n players with $n \geq 3t + 1$, at most any t players may be corrupt. Suppose D be the Dealer. D wants to share a secret $S \in F$ where F is a field.

NOTE: All computation will be over this field F

Now there can be at most t corrupted players. Total number of subsets with t players are $\binom{n}{t} = k$ (say). Suppose $S_1, S_2, ...S_k$ are these subsets. Now choose k numbers from the field F such that $S = r_1 + r_2 + ... + r_k$. We can do it easily as we can generate random numbers $r_1, r_2, ...r_{k-1}$ and then let r_k= S-$(r_1 + r_2 + ... + r_{k-1})$. This is also called *additive sharing*. Now dealer D gives r_l share to each player who are not in S_l, l=1,2....,k. Therefore at least $n - t$ player get r_l share and these t players who are in S_l will never get r_l also among these at least $n - t = 2t + 1$ players, at least t+1 players are honest. Now in reconstruction phase we want to reconstruct S for this we need r_l, $l = 1, 2, ..., k$ shares. So for r_l share every player who is not in S_l, will give his r_l share. Among them at most t can be corrupted and at least $t + 1$ will be honest. So take majority based decision for r_l. Now we need to think more, suppose If dealer D is dishonest then we need to ensure that all honest players get same share. That is say dealer shared r_l among all $n - t$ players who are not in S_l, then it should be insure that all honest players get same r_l. Question is- *but why we need to ensure it ?* otherwise there will be problem in reconstruction phase. since then we cannot take decision on majority based. Another question is- *but how we will ensure it ?* for this each pair of players p_i and p_j need to exchange all their common shares. (because D shares r_l using private channel so p_i and p_j do't know what p_i get and what p_j get) to check if they agree. If they disagree. a complaint is being broadcasted. If there is complaint then either the dealer shared bad value or the dealer is honest and bad player is exchanging incorrect values. To settle this ambiguity, dealer has to reveal the controversial shares.

NOTE : This information is already in the hands of adversary so no extra information adversary get.
NOTE : If a bad dealer modifies the controversial share at this point there is still a unique value committed as the shared secret because the secret is defined as sum of the k shares.

4.2 Protocol:

Sharing phase:

- *Round* (1) Let $S_1, S_2, ...S_k$ be the sets of all possible subsets with t players, dealer D has a secret $S \in F$ (field), D choose k random values $r_1, r_2, ...r_k \in$F such that $S = r_1 + r_2 + ... + r_k$.
    1. D sends to player P_i the value r_l such that $P_i \notin S_l$, $\{l = 1, 2, ...k\}$
    2. Simultaneously, each player P_i sends to each player P_j a random paid $r^l_{ij} \in$F and $P_i, P_j \notin S_l$, $\{l = 1, 2, ...k\}$.
- *Round* (2) If $P_i, P_j \notin S_l$,

1. P_i broadcasts $a_{ij}^l = r_l + r_{ij}^l$
2. P_j broadcasts $a_{ji}^l = r_l + r_{ij}^l$, $\forall\ l = 1, 2, ..., k$.

- *Round* (3) For each pair P_i and P_j such that $a_{ij}^l \neq a_{ji}^l$ the dealer broadcasts the value r_l which is now taken by all the players.

Reconstruction Phase:
For shares r_l each player $P_i \notin S_l$ provides the share r_l it owns. Take the value that appears most often as the proper share r_l. Set $Rec = r_1 + r_2 + ... + r_k$.

Theorem 4.2.1 *The three round protocol satisfies all the three properties of VSS.*

Proof We will show that all the three properties satisfies by the protocol. As we know that the *secrecy* and the *correctness* properties make sense, if adversary is *honest*. Third property *strong commitment* is required to show when dealer is *dishonest*. So we will take two cases.

- *HONEST DEALER:*

  Secrecy: We need to show that the secrecy is preserved up to sharing phase. For this we analyse each round of sharing phase. In *Round (1)* we note that among $S_1, S_2, ...S_k$ there is at least one set that contains all corrupted players. Say S_l be that set. Now dealer gives r_l share to those players who are not in S_l. Therefor there is one share that adversary does not get. So secrecy is preserve in round (1). In *Round* (2) players broadcast their shares with random pad (mask), so no information is revealed about the secret. In *Round* (3) dealer broadcasts controversial share for $a_{ij}^l \neq a_{ji}^l$. This value is already known to adversary. So no new information revealed.

  Correctness: We have to show that if dealer is honest then in reconstruction phase S will be reconstruct, irrespective of the behavior of adversary. It is obvious because for every share r_l, we are taking majority based decision and every time we can get only at most t corrupted shares and at least $t + 1$ share will be correct. Therefore we can reconstruct the same secret S.

- *DISHONEST DEALER:*

  Strong commitment: we need to show that if dealer is corrupted then there exits some $S^* \in$F such that S^* is commited/shared by dealer during sharing phase and same S^* will be reconstructed during reconstruction phase irrespective of the behavior of adversary. Since the secret is defined as the sum of the k shares. So even if D is dishonest, then still a unique value committed as the shared secret at the end of the sharing phase. Hence the property of strong commitment is satisfied.

4.3 Conclusion

In the present chapter we studied an inefficient 3-round VSS protocol [5] with $n \geq 3t+1$, using additive sharing. In the next chapter we will discuss an efficient 4-round VSS protocol with $n \geq 3t + 1$.

Chapter 5

Efficient Four-Round VSS Protocol with $n \geq 3t + 1$

In this chapter we will discuss an efficient four round protocol of [5] with $n \geq 3t + 1$. Suppose $P = \{P_1, P_2, ...P_n\}$ be the set of n players and t be the threshold value.

5.1 Idea of the Protocol:

We will use the concept of bivariate polynomial. Let us suppose that $P_1, P_2, ...P_n$ be the n-players with $n \geq 3t + 1$ and among these n-players at most t players may be corrupted. Suppose dealer D is any one of them. Now we think about every situation and try to design a protocol.

1. Dealer D chooses a random bivariate polynomial $F(x, y)$ of degree t in x and y such that $F(0, 0) = S$ be the secret. D sends to the player P_i the polynomials $f_i(x) = F(x, i)$ and $g_i(y) = F(i, y)$ privately. At this point question is- *Can dealer broadcast these univariate polynomials?* No because if D broadcasts, then there is no meaning of secrecy, adversary can get secret easily. Both $f_i(x)$ and $g_i(y)$ are univariate polynomials of degree t. But think what is the guaranty that D has send right (consistent) $f_i(x) = F(x, i)$ and $g_i(y) = F(i, y)$. That is if D is dishonest then D can send inconsistent $f_i(x)$ and $g_i(y)$. So we need to check it. *But how can we check it?* Since only P_i knows about it's $f_i(x)$ and $g_i(y)$. So there is need for communication between players, and take the benefit of property of the bivariate polynomials.

2. Each player P_i sends to P_j the value $g_i(j)$ privately $\forall\ j = 1, 2, ...n$. Now again the same question. *why P_i sends value privately-?* because if P_i broadcast the value $g_i(j)$, then it will be known to adversary and adversary can reconstruct the secret. But *what is the benefit of sending $g_i(j)$?* Since

now P_j can check whether $f_j(i) = g_i(j)$ or not. If yes then both P_i and P_j are consistent.

3. Each player P_j broadcasts a list L_j of players P_i for whom it holds that $g_i(j) \neq f_j(i)$. Now every player knows that where is inconsistency. Suppose P_i and P_j are inconsistent. So either P_i or P_j is corrupt or dealer has shared inconsistent $g_i(y)$ or $f_j(x)$. So at this point we need to know what is the controversial share $F(i, j)$.

4. Dealer broadcasts $F(i, j)$. But still there is no guaranty that this $F(i, j)$ broadcasted by D is right share. So there is need to give chance to unhappy players, say P_i for complain against this value.

5. Player P_i may complain if this value which was revealed by the dealer is inconsistent with its own private information. Now again it may possible that D is honest. That is, dealer had broadcast right share and player P_i is corrupt. So *how we will come to know this?* Yes there is one way, if D broadcasts $f_i(x)$ and $g_i(y)$ related to P_i then every one can know that broadcasted $F(i, j)$ by D was consistent or not.

6. Dealer D broadcasts all the information held by the P_i. That is D broadcasts $f_i(x)$ and $g_i(y)$. Now again there is case P_i is honest and $f_i(x)$ and $g_i(y)$ are inconsistent. But since $f_i(x)$ and $g_i(y)$ are public so players can check $f_i(j) = g_j(i)$ or not. If not, again we need to complain.

7. Complaints from the players.

But we notice, that this is a 7*-round protocol. But we want to design a* 4*- round protocol. So we need to think that can we compressed the rounds. First* 3*-rounds can be compressed to* 2*-rounds using the concepts of random pad.*

1. Dealer D sends to P_i the polynomials $f_i(x)$ and $g_i(y)$. Simultaneously P_i sends to P_j an independent random pad r_{ij} use uniformly from the field F.

2. Player P_i broadcasts
   $a_{ij} = f_i(j) + r_{ij}$
   $b_{ij} = g_i(j) + r_{ji}$
   We note that, $a_{ij} = f_i(j) + r_{ij} = F(j, i) + r_{ij}$
   and $b_{ji} = g_j(i) + r_{ij} = F(j, i) + r_{ij}$
   That is if P_i and P_j are consistent then $a_{ij} = b_{ji}$. But if $a_{ij} \neq b_{ji}$ then there is inconsistency between P_i and P_j. But dealer now cannot tell whether P_i or P_j is misbehaving. Because shares of P_i and p_j are masked and to know the real shares, D need to know the unmasked controversial share.

3. For each pair $a_{ij} \neq b_{ji}$
   P_i broadcasts $\alpha_{ij} = f_i(j) = F(j,i)$
   P_j broadcasts $\beta_{ji} = g_j(i) = F(j,i)$
   D broadcasts $\gamma_{ij} = F(j,i)$
   after this stage D knows that which player is misbehaving say P_i. So instead broadcast its real share $F(i,j)$, dealer D should broadcast $f_i(x)$ and $g_i(y)$. So that we can compressed one more round.

4. Dealer D broadcasts $f_i(x)$ and $g_i(y)$. Now again there is need to verify this public polynomial, because if D is corrupted then there is chance that broadcasted $f_i(x)$ and $g_i(y)$ may not be consistent.

5. Players broadcast a complaint if necessary.

Still we compressed two rounds, but to achieve 4*-round protocol we need to think more. We think about steps* $3, 4$ *and* 5*. In step* 3 *players broadcast their shares, and in step* 4 *dealer broadcast* $f_i(x)$*, and in step* 5 *need to verify* $f_i(x)$*. But if* D *broadcasts controversial share* $F(j,i)$ *in step* 3 *then every one know which player is misbehaving. So in step* 4 *dealer broadcasts* $f_i(x)$ *and each player broadcast in the clear his point on* $f_i(x)$*. Everybody now can check whether point is consistent or inconsistent.*

5.2 Protocol:

Sharing phase:

- *Round* (1) Dealer D chooses a random bivariate polynomial $F(x,y)$, over a field F of degree t in x and y with $F(0,0) = S$ be the secret.

  1. D sends to player P_i the polynomials $f_i(x)$ and $g_i(y)$
  2. Player P_i sends to player P_j an independent random pad r_{ij} choose from field F.

- *Round* (2) Player P_i broadcasts,

  1. $a_{ij} = f_i(j) + r_{ij}$
  2. $b_{ij} = g_i(j) + r_{ji}$.

- *Round* (3) For each pair $a_{ij} \neq b_{ji}$,

  1. P_i broadcasts $\alpha_{ij} = f_i(j)$
  2. P_j broadcasts $\beta_{ji} = g_j(i)$
  3. D broadcasts $\gamma_{ij} = F(j,i)$
     a player is said to be *unhappy* if the value which he broadcasts does not match with the dealer's value. If there are more than t unhappy players then disqualified the dealer.

- *Round* (4) For each *unhappy* player P_i, dealer broadcasts the polynomial $f_i(x)$ and each *happy* player P_j broadcasts $g_j(i)$.

Local computation: For each public polynomial $f_i(x)$, check for at least $2t+1$ happy players P_j, the following holds $g_j(i) = f_i(j)$. If it is not the case than disqualify the dealer

Reconstruction Phase:

We note that $F(0,0) = S$, and constant term of $g_0(y) = F(0,y)$ is the secret S. Also note that $f_i(0) = g_0(i)\ \forall i = 1, 2, ...n$. So every player P_i provides $f_i(0)$, so we get n points on a polynomial among these n points at most t may be wrong. Now use *Reed-Solomon error-correction* [13] algorithm and get polynomial $g_0(y)$ and secret $S = g_0(0)$.

5.3 Conclusion

In this chapter we discussed an efficient 4-round protocol with $n \geq 3t+1$ given by Gennaro et. al [5]. In the next chapter we will study a primitive of VSS, called WSS. Also in the next chapter we will propose a 1-round WSS protocol for $n > 4t$ with communication complexity $O(n)$.

Chapter 6

Weak Secret Sharing

Weak secret sharing (WSS) is a variant of verifiable secret sharing, where the reconstructed value may also be some default value, in case the dealer is corrupted.

6.1 Weak Secret Sharing

Let $P = \{P_1, P_2, ...P_n\}$ be the set on n players, and D be the dealer. Now there are two models:

1. $D \in P$
2. $D \notin P$.

We will assume $D \in P$ and also assume that at most t players can be actively corrupted (possibly including D). Now we define the weak secret sharing. WSS is a two phase protocol (Sharing and Reconstruction) carried out among n parties in the presence of a active adversary. The goal of the WSS is to share a secret, S, among the n parties during the sharing phase in a way that would later allow for a unique reconstruction of this secret in the reconstruction phase, while preserving the secrecy of S until the reconstruction phase.

Sharing Phase: The dealer initially holds secret $S \in F$ where F is a finite field of sufficient size; and each player P_i holds some private information r_i. The sharing phase may consist of several rounds. At each round, each player can privately send messages to other players or can broadcast a massage.

Reconstruction Phase: In the reconstruction phase, each player P_i revealed information r_i^* (a dishonest player may reveal $r_i^* \neq r_i$), a reconstruction function Rec is applied in order to compute the secret, $S = Rec(v_1^*, ..., v_n^*)$.

6.1.1 Properties of WSS

Any WSS protocol should satisfy the following properties:

Secrecy: If D is *honest* (i.e., is uncorrupted by the adversary), then the adversary's view during the sharing phase reveals no information about S. More formally, the adversary's view is identically distributed under all different values of S. It means, if D is honest, then secret S is perfectly secure during the sharing phase.
NOTE: We need secrecy, if D is honest and up to the sharing phase.

Correctness: If D is *honest*, then the reconstructed value is always equal to the secret S, irrespective of the behavior of adversary.

Weak commitment: If D is *dishonest*, then at the end of the sharing phase there is a unique value $S^* \in F$, such that at the end of the reconstruction phase all honest players output S^* or default value $\perp \notin F$, irrespective of the behavior of the corrupted players.

6.2 VSS versus WSS

WSS is a primitive which satisfies the same properties as VSS except for the commitment property. VSS has a strong commitment, which requires that at the end of the sharing, there is a fixed value S^* and that the honest parties output this value in the reconstruction phase. In contrast, WSS has a weaker commitment property which requires that at the end of the reconstruction phase, the honest parties output S^* or default value $\perp \notin F$.

6.3 Environment for WSS Protocols

We follow the environment for WSS protocols same as we presented in section 2.6 of Chapter 2. Recall that the set of players is denoted by $P = \{P_1, P_2, ...P_n\}$, and adversary is denoted by A_t.

Theorem 6.3.1 *One-round as well as two-round perfect WSS protocol, tolerating adversary A_t, is possible if and only if $n \geq 4t + 1$.*

Theorem 6.3.2 *Three-round WSS protocol, tolerating adversary A_t, is possible if and only if $n \geq 3t + 1$.*

6.4 1-Round WSS Protocol for $n > 4t$

Fitzi et al. [1] designed a 1-round WSS protocol, in which they used a random bivariate polynomial.

Theorem 6.4.1 *There is a 1-round WSS protocol which satisfies all the three properties of WSS., with communication complexity $O(n^2)$.*

6.5 Our 1-Round WSS Protocol for $n > 4t$

Here we design a 1-round WSS protocol for $n > 4t$. It is shown that there exists a 1-round WSS protocol for $n > 4t$ by Fitzi et al. [1] They used the concept of bivariate polynomial.

6.5.1 Idea of the Protocol

In our 1-round WSS protocol, instead of bivariate polynomial, we will use univariate polynomial. To reconstruct the secret, we will use *Reed-Solomon error correction* algorithm [12]. So first we state the Reed-Solomon error correction.

6.5.2 Reed-Solomon error correction

Suppose we have given n distinct points of which at least $n - t$ points are on some unique t degree polynomial, say $f(x)$. That is we know out of n points, at most t are wrong. It means at most t points do not lie on the polynomial $f(x)$. But we do not know which t points are wrong and what is $f(x)$. Now our goal is to reconstruct the polynomial $f(x)$ from the given n points. To solve this problem we have the following theorem:

Theorem 6.5.1 *[12] Reed-Solomon error correction could reconstruct $f(x)$ from the n points, even if t of them are corrupted if and only if $n \geq 3t + 1$.*

We now present the protocol. The secret S is assumed to be taken from a finite field F with $|F| > n$, additionally, $1, 2, ..., n$ are interpreted as distinct non-zero field elements. Suppose $P = \{P_1, P_2, ..., P_n\}$ be the set of n players and A_t be the adversary.

6.5.3 Protocol

- *Sharing Phase:* One Round
  Dealer D chooses a random univariate polynomial $f(x)$ of degree t, over the field F, with $f(0) = S$ be the secret.
  - D sends privately to the player P_i the value $f(i)$ as a share.
- *Reconstruction Phase:*
  Every player P_i broadcasts his share $f(i)$. Now to reconstruct the secret apply Reed-Solomon error correction algorithm. There arise two cases:
  - *Case (1)*: If we get a polynomial $f^*(x)$ of degree t, after applying the Reed-Solomon error correction algorithm, then secret is $S^* = f(0)$. That is secret is the constant term of the constructed polynomial.
  - *Case (2)*: If we do not get a polynomial of degree t, after applying the Reed-Solomon error correcting algorithm, then reconstruct a default value $\perp \notin F$

Theorem 6.5.2 *Our* 1*-round* WSS *protocol satisfies, all the three properties of* WSS *protocol.*

Proof We will show that all the three properties of the WSS satisfies by the protocol. As we know that the *secrecy* and the *correctness* properties make sense, if adversary is *honest*. Third property *weak commitment* is required to show, when dealer is *dishonest*. So we will take two cases.

- *HONEST DEALER:*

  Secrecy: We need to show that the secrecy is preserved up to the sharing phase. In sharing phase D sends privately the value $f(i)$ to the player P_i. So even if adversary controls over t players, then the adversary has knowledge of t values on the polynomial. But to reconstruct the polynomial of degree t, adversary needs at least $t + 1$ values on the polynomial. Hence secrecy is preserved.

  Correctness: It is straight forward. Dealer is honest then he sends correct shares to all the players. Thus at the end of the reconstruction phase, using Reed-Solomon error correction algorithm, we always reconstruct the same secret. which was shared by the dealer in the sharing phase.

- *DISHONEST DEALER:*

  Weak Commitment: Now we have to consider D is corrupted. As we know in our protocol, there are two cases in the reconstruction phase.

If *case (2)* arise then all the players compute $\perp$ and weak commitment is satisfied. On the other hand, what happened, if *case* 1 arise. That is a t degree polynomial constructed in the reconstruction phase. We will show that, if in the reconstruction phase a t degree polynomial exists, then it will be the committed polynomial by the dealer. *We know that two polynomials of degree t can have at most t common values.* That is the two polynomials of degree t can cut each other at most t points. Now suppose adversary shares two secrets S_1 and S_2, corresponding to, two t degree polynomials $f_1(x)$ and $f_2(x)$ respectively. Now at most t players can be dishonest. A dishonest player can produce any value in the reconstruction phase. Remaining $n-t$ players are honest. Among these $n-t$ honest players at most t players can have mixed shares (because two polynomials of degree t can have at most t common values). Now suppose D shares S_1 with t honest players and S_2 with other t honest players. *Note:If D shares* S_i $\{i = 1 or 2\}$ *more than t honest players then* S_i *will be the committed secret or* $\perp$ *will be reconstructed.* Now almost we have done. Actually, now we have t corrupted players, t honest players with shares of S_1, t honest players with shares of S_2, and t honest players with mixed shares of S_1 and S_2. So we have taken care of $4t$ players, but our protocol works for $n > 4t$ players. Without loss of generality assume that, there are $n = 4t+1$ players. Now the reconstruction of the polynomial will depend on the share of the remaining one honest player. Without loss of generality suppose this remaining player has share of S_1. Now if corrupted players give shares of S_1 in the reconstruction phase, then we have $3t+1$ players with shares of S_1. S_1 will be the committed secret. On the other hand, if the corrupted players give shares of S_2 in the reconstruction phase, then a default value $\perp$ reconstruct. But S_2 will never reconstruct. So always unique value will be reconstruct. Hence weak commitment property holds good.

6.5.4 Comparison between existing and our 1-Round WSS Protocol

Existing 1-round WSS protocol of Fitzi et. al [5] has *communication complexity* $O(n^2)$. In our proposed WSS protocol, during the sharing phase the dealer D sends only one value to each player P_i. As there are n players, therefore the total *communication complexity* of the proposed WSS protocol will be of $O(n)$, which is significantly less than the existing 1-round protocol.

Chapter 7

conclusion

In this Thesis we discussed three important tools in cryptology namely, *Secret Sharing, Verifiable Secret Sharing* and *Weak Secret Sharing*. We also studied some existing perfect VSS protocol. In this work we proposed a 1- round WSS protocol with *communication complexity* $O(n)$. Now there are some open problems:

- *Open Problem* (1)*:* To derive lower bound on *communication complexity* of perfect VSS.
- *Open Problem* (2)*:* To design a computationally efficient 4 round statistical VSS with $n = 2t + 1$.

Bibliography

[1] M. Fitzi, J. Garay, S. Gollakota, C. Pandu Rangan, and K. Srinathan. Round-Optimal and Efficient Verifiable Secret Sharing. In S. Halevi and T. Rabin, editors, *Theory of Cryptography, Third Theory of Cryptography Conference, TCC* 2006*, New York, NY, USA, March* $4-7, 2006$*, Proceedings, volume* 3876 *of Lecture Notes in Computer Science, pages* $329-342$. Springer Verlag, 2006.

[2] B. Chor, S. Goldwasser, S. Micali, and B. Awerbuch. Verifiable Secret Sharing and Achieving Simultaneity in the Presence of Faults (Extended Abstract). *In Proceedings of the 17th Annual ACM Symposium on Theory of Computing, May* $6-8$*,* 1985*, Providence, Rhode Island, USA, pages 383-395.* ACM Press, 1985.

[3] M. Tompa and H. Woll. How to Share a Secret with Cheaters. *Journal of Cryptology*, 1(2):$133-138$, 1988.

[4] R. J. McEliece and D. V. Sarwate. On Sharing Secrets and Reed-Solomon Codes. *Communications of the ACM*, 24(9):$583-584$, 1981.

[5] R. Gennaro, Y. Ishai, E. Kushilevitz, and T. Rabin. The Round Complexity of Verifiable Secret Sharing and Secure Multicast. *In Proceedings on* 33*rd Annual ACM Symposium on Theory of Computing, July* $6-8$*,* 2001*, Hersonissos, Crete, Greece. ACM*, pages $580-589$. ACM Press, 2001.

[6] Arpita Patra. *Studies on Verifiable Secret Sharing, Byzantine Agreement and Multiparty Computation.* PhD thesis, 2010 Indian Institute of Technilogy, Madras, Chennai, India.

[7] Ashish Choudhury. *Protocols for Reliable and Secure Message Transmission.* PhD thesis, 2010 Indian Institute of Technilogy, Madras, Chennai, India.

[8] D.R. Stinson, *An Explication of Secret Sharing Schemes*, Designs, Codes and Cryptography, 2, $357-390$ (1992) @ 1992 *Kluwer Academic Publishers. Manufactured in The Netherlands.*

[9] William Stallings - *Cryptography and Network Security*, Fourth edition, Pearsons Education-2006.

[10] F.J. MacWilliams, N.J.A Sloane - *The Theory of Error-Correcting Codes, Part* -1, Bell Laboratories Murray Hill NJ 07974, U.S.A.

[11] Douglas R. Stinson - *Cryptography Theory and Practice*, Third edition, Chapman and Hall/CRC, New York.

[12] Irving S. Reed, Gustave Solomon - *Polynomial Codes over Certain Finite Fields*, Journal of the Society for Industrial and Applied Mathematics, 8(2): $300 - 304$, 1960.

[13] T. Rabin and M. Ben-Or. Verifiable Secret Sharing and Multiparty Protocols with Honest Majority (Extended Abstract). In *Proceedings of the* 21^{st} *Annual ACM Symposium on Theory of Computing*, May $14 - 17$, 1989, Seattle, Washigton, USA, *pages* $73 - 85$. ACM Press, 1989.

[14] D. Dolev, C. Dwork, O. Waarts, and M. Yung. *Perfectly Secure Message Transmission*. Journal of ACM, $40(1) : 17 - 47, 1993$.