Western Kentucky University
TopSCHOLAR®

Masters Theses & Specialist Projects | Graduate School

5-1-2012

Efficient algorithm to construct phi function in vector space secret sharing scheme and application of secret sharing scheme in Visual Cryptography

Sunny Potay
Western Kentucky University, sunny.potay664@topper.wku.edu

Follow this and additional works at: http://digitalcommons.wku.edu/theses

Part of the Theory and Algorithms Commons

Recommended Citation
Potay, Sunny, "Efficient algorithm to construct phi function in vector space secret sharing scheme and application of secret sharing scheme in Visual Cryptography" (2012). *Masters Theses & Specialist Projects.* Paper 1151.
http://digitalcommons.wku.edu/theses/1151

This Thesis is brought to you for free and open access by TopSCHOLAR®. It has been accepted for inclusion in Masters Theses & Specialist Projects by an authorized administrator of TopSCHOLAR®. For more information, please contact connie.foster@wku.edu.

EFFICIENT ALGORITHM TO CONSTRUCT FUNCTION IN VECTOR SPACE SECRET SHARING SCHEME AND APPLICATION OF SECRET SHARING SCHEME IN VISUAL CRYPTOGRAPHY

A Thesis
Presented to
The Faculty of the Department of Mathematics and Computer Science
Western Kentucky University
Bowling Green, Kentucky

In Partial Fulfillment
Of the Requirements for the Degree
Master of Science

By
Sunny Potay

May 2012

EFFICIENT ALGORITHM TO CONSTRUCT ϕ FUNCTION IN VECTOR SPACE SECRET SHARING SCHEME AND APPLICATION OF SECRET SHARING SCHEME IN VISUAL CRYPTOGRAPHY

Date Recommended April, 25, 2012

Dr. Mustafa Atici, Director of Thesis

Dr. James Gary

Dr. Qi Li

Dean, Graduate Studies and Research Date 18-May-2012

ACKNOWLEDGMENTS

It was a great pleasure working under the guidance of Dr. Mustafa Atici for the success of this thesis. I would like to thank Dr. Mustafa Atici for the immense trust and patience he has over me.

I would like to thank Dr. James Gary and Dr. Qi Li for their valuable suggestions to improve this thesis.

I would like to thank my family for their immense never ending support.

TABLE OF CONTENTS

1. INTRODUCTION **01**

1.1. Fine set of Z_p and its properties 01

1.2. Cryptosystems 03

1.3. Secret Sharing Cryptosystem 08

1.4. Some basic definition 09

2. SCHEMES **11**

2.1. Shamir Threshold Scheme 12

2.2. Monotonic Circuit Construction Scheme 16

2.3. Brickell Vector Space Construction Scheme 22

3. PHI FUNCTION **28**

3.1. Construction of function for special access structure I 30

3.2. Construction of function for special access structure II 34

3.3. Brute force search 43

4. VISUAL CRYPTOGRAPHY **46**

4.1. Generating permutation from secret key K 51

5. CONCLUSION **55**

LIST OF FIGURES

Figure 1. Illustration of encryption and decryption using a secret key 05

Figure 2. Illustration of authorized sets 12

Figure 3. Monotone share distribution to set of participants 21

Figure 4. Bipartite Graph G 28

Figure 5. Complete Bipartite Graph G: $K_{4,3}$ 29

Figure 6. Multipartite Graph G 29

Figure 7. Complete Multipartite Graph G: $K_{2,3,4}$ 30

Figure 8. Complete Multipartite Graph G: $K_{3,3}$ 31

Figure 9. Multipartite graph representing Γ 35

Figure 10. Visual cryptography 47

Figure 11. Original Image 49

Figure 12. Encrypted Image 50

Figure 13. Decrypted Image 50

Figure 14. Image Encrypted 51

Figure 15. Image Decrypted 51

LIST OF TABLES

Table 1. Additive Inverse 02

Table 2. Multiplicative Inverse 03

Table 3. System Configuration used for Brute Force 43

Table 4. Programming languages and tools used for Brute Force 44

EFFICIENT ALGORITHM TO CONSTRUCT FUNCTION IN VECTOR SPACE SECRET SHARING SCHEME AND APPLICATION OF SECRET SHARING SCHEME IN VISUAL CRYPTOGRAPHY

Sunny Potay May 2012 57 Pages

Directed by: Dr. Mustafa Atici, Dr. James Gary, Dr. Qi Li

Department of Mathematics and Computer Science Western Kentucky University

Secret Sharing refers to a method through which a secret key K can be shared among a group of authorized participants, such that when they come together later, they can figure out the secret key K to decrypt the encrypted message. Any group which is not authorized cannot determine the secret key K. Some of the important secret schemes are Shamir Threshold Scheme, Monotone Circuit Scheme, and Brickell Vector Space Scheme. Brikell's vector space secret sharing construction requires the existence of a function from a set of participant $\mathcal{P}$ in to vector space Z_p^d, where p is a prime number and d is a positive number. There is no known algorithm to construct such a function in general. We developed an efficient algorithm to construct function for some special secret sharing scheme. We also give an algorithm to demonstrate how a secret sharing scheme can be used in visual cryptography.

CHAPTER 1: INTRODUCTION

In this chapter, we will walk through all the general definitions which are used throughout the thesis. Moreover, basic concepts like cryptosystems are covered with illustrative examples.

1.1 Finite set Z_p and its properties

Let p be a positive integer. Then we define $Z_p = \{0, 1, \dots, p-1\}$. Suppose a, b are two integers. Then $a \equiv b \; mod \; p$ if and only if $(a - b)$ is evenly divisible by p, that is $a - b = k \times p$ for some integer k.

Example 1.1.1: Assume a value $a = 23$ which exceeds modulo $p = 17$ and let $b = 8$ then $23 \equiv 8 \; mod \; 17$, since $23 - 8 = 17$.■

Properties of Finite set Z_p [1, 13]

- Addition is closed, i.e., for any $a, b \in , Z_p \; a + b \in Z_p$.
- Addition is commutative, i.e., for any $a, b \in Z_p, a + b = b + a$.
- Addition is associative, i.e., for any $a, b, c \in Z_p, (a + b) + c \; = \; a + \; (b + c)$.
- Additive Identity, i.e., for any $a \in Z_p, a + 0 = 0 + a = a$.
- Additive Inverse, i.e., for any $a \in Z_p$, there exists a unique b such that $a + b = b + a = 0 \; mod \; p$.
- Multiplication is closed, i.e., for any $a, b \in Z_p, ab \in Z_p$.

- Multiplication is commutative, i.e., for any $a, b \in Z_p, ab = ba$.

- Multiplication is associative, i.e., for any $a, b, c \in Z_p$ $a(bc) = (ab)c$.

- Multiplicative Identity, i.e., for any $a \in Z_p$ $a \times 1 = 1 \times a = a$.

- Multiplicative Inverse, i.e., for any non-zero $a \in Z_p$ there exist a^{-1} such that

  $a \times a^{-1} \equiv 1 \; mod \; p$. In order to have multiplicative inverse p must be a prime number.

Example 1.1.2: Illustration of additive inverse

+	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
0	0	1	2	3	4	5	6	7	8	9
1	1	2	3	4	5	6	7	8	9	0
2	2	3	4	5	6	7	8	9	0	1
3	3	4	5	6	7	8	9	0	1	2
4	4	5	6	7	8	9	0	1	2	3
5	5	6	7	8	9	0	1	2	3	4
6	6	7	8	9	0	1	2	3	4	5
7	7	8	9	0	1	2	3	4	5	6
8	8	9	0	1	2	3	4	5	6	7
9	9	0	1	2	3	4	5	6	7	8

Table-1 Additive inverse table■

Example 1.1.3: Illustration of multiplicative inverse

*	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
0	0	0	0	0	0	0	0	0	0	0
1	0	1	2	3	4	5	6	7	8	9
2	0	2	4	6	8	0	2	4	6	8
3	0	3	6	9	2	5	8	1	4	7
4	0	4	8	2	6	0	4	8	2	6
5	0	5	0	5	0	5	0	5	0	5
6	0	6	2	8	4	0	6	2	8	4
7	0	7	4	1	8	5	2	9	6	3
8	0	8	6	4	2	0	8	6	4	2
9	0	9	8	7	6	5	4	3	2	1

Table-2 Multiplicative inverse table■

1.2 Cryptosystems

Cryptography is the practice and study of techniques for secure communication and hiding of data in the presence of third parties. The fundamental objective [2,13] of cryptography is to enable two people to communicate over an insecure channel and to hide data without the risk of a third person interrupting it. Consider two people, Alice and Bob, who want to communicate (Email, credit card payment, chat messenger, etc.) over different insecure media. John is a third person who wants to tap the communication between Alice and Bob for overhearing or fabricating data between them so he might misuse it just like in credit cards. This is possible because the data is vulnerable on channel as it is in readable manner. So to secure the communication, Alice and Bob have to hide the data from others in such a way that the data is not readable. This can be achieved by Alice sending a message X to Bob by encrypting the message. One of the very early encryption methods is shifting characters by K position in X. For example, message X=WORD and shift value K=2. Then W is shifted to Y, O is shifted to Q, R is shifted to T and D is shifted to F. Hence message X=WORD is changed to YQTF. This

general process is called an encryption. Encryption is defined as a technique for hiding of plain text or message(X) by using secret key K. This encrypted message is no longer meaningful to John. But when Bob receives the encrypted message he would not be able to read it without knowing secret key K. So for secure communication between Alice and Bob, they both should agree on a shift value K. Alice can use to encrypt the message and Bob can use it to decrypt, which is reverse shifting with K on encrypted message i.e., Y is shifted to W, Q is shifted to O, T is shifted to R and F is shifted to D. This process is called a decryption. Decryption is defined as a technique for retaining back the original message from the encrypted one.

Definition1.2 [2,13]: Cryptosystem is a 5-tuple $(\mathcal{P}, \mathcal{C}, \mathcal{K}\ ,, \mathfrak{D})$, where

$\mathcal{P}$ is a finite set of plain text to create a message,

$\mathcal{C}$ is a finite set of cipher text to create an encrypted message,

$\mathcal{K}$ is a key space of finite set of possible keys K, which are used for encryption,

is defined as a set of encryption rules, and

$\mathfrak{D}$ is defined as a set of decryption rules.

For each $K \in \mathcal{K}$, there is an encryption rule $e_K \in \mathcal{E}$ and a decryption rule $d_K \in \mathfrak{D}$, where $e_K: \mathcal{P} \to \mathcal{C}$ and $d_K: \mathcal{C} \to \mathcal{P}$ are functions such that $d_K\big(e_K(\mathrm{X})\big) = \mathrm{X}$ for every plaintext $\mathrm{X} \in \mathcal{P}$.

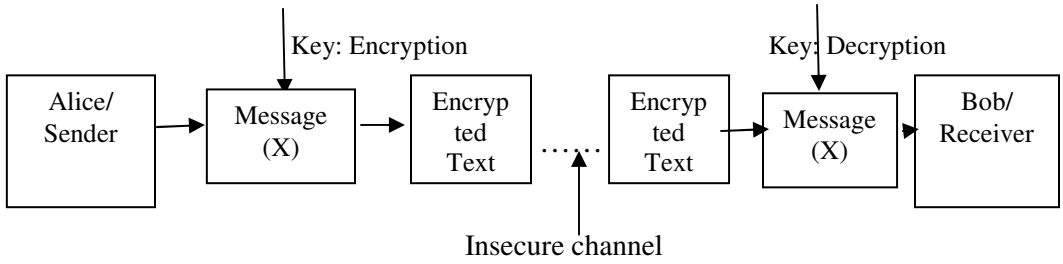

Figure-1 Illustration of encryption and decryption using a secret key

The above Figure-1 illustrates that the sender encrypts the plain text with a secret key and passes it on to an insecure channel. The receiver on the other end receives the encrypted text and decrypts it with the same secret key to recover the secret plain text sent by the sender. Meanwhile no other third party can recover the text unless they have the same secret key (K). This is illustrated by using two brief examples.

Example 1.2.1: Illustration of Shift Cipher [8] as of cryptosystem. Shift Cipher consists of 5-tuple $< Z_{26}, Z_{26}, \mathcal{K}, , \mathfrak{D} >$, where $\mathcal{P} = \mathcal{C} = \mathcal{K} = Z_{26} = \{0, \dots, 25\}$. Encryption and decryption rules are defined as follows: Let K be a secret key from key space $\mathcal{K}$ and X is message from plain text $\mathcal{P}$:

$$e_K(\mathrm{X}) = (\mathrm{X} + K) \, mod \, p \, .$$

$$d_K\big(e_K(\mathrm{X})\big) = (\, e_K(\mathrm{X}) - K) \, mod \, p \, .$$

A	B	C	D	E	F	G	H	I
0	1	2	3	4	5	6	7	8

J	K	L	M	N	O	P	Q	R	S
9	10	11	12	13	14	15	16	17	18

T	U	V	W	X	Y	Z

19	20	21	22	23	24	25

A↔0, …, Z↔25 is represented in the above table. Consider Alice as the sender, Bob as the receiver and John as a third person. Now Alice wants to send a message "MEET YOU AT ONE" to Bob but he doesn't want anyone to overhear or fabricate the original message on the way to Bob. Alice and Bob share a secret key which is private only to them, say K=2. Alice encrypts the message using Shift cipher technique by shifting the character in message X by 2. Before she encrypts the white spaces are removed from message X.

Message X= MEETYOUATONE.

Transform the characters in message X into integers, from the above table. Now the secret key K is added to the integer representing each character in message X. If the sum exceeds 26, say 29. Then the sum would be 3 after being reduced by modulo 26.

M: = (12+2) mod 26= 14, gives you O based on the above table.

E: = (4+2) mod 26= 6, gives G.

Similarly the rest of the cipher characters are computed.

14	20	21	22	23	24	25	2	21	16	15	6

The Cipher text is based on the sum values generated from the above table. The message which John receives if he taps the channel is OGGVAQWCVQPG; this is non-understandable to him. So the message is secured and safely delivered to Bob. Bob uses

the same secret key K=2 to decrypt it by shifting it back by two characters to transform cipher text into "the original plain text" (without white spaces).■

Example 1.2.2 Permutation Crypto system is 5-tuple $\langle Z_{26}, Z_{26}, \mathcal{K}, , \mathfrak{D}\rangle$ where plain text space and cipher text space are Z_{26} and key set $\mathcal{K}$ is set of all permutation over $\{1,2,\ldots,26\}$. This technique is different from shift cipher as shift cipher uses substitution, whereas permutation cipher replaces plain text with different cipher text characters. Let's say Alice wants to send a message X to Bob which is X=MEET AT ONE. Now the secret key is the permutation which replaces the character of plain message X with cipher characters. The Encryption and decryption method for permutation cipher is as follows

$$e_{\Pi}(X) = K(X).$$

$$d_{\Pi}\big(e_{\Pi}(X)\big) = K^{-1}\big(e_{\Pi}(X)\big).$$

Let's pick secret permutation K as key, follows: e_{Π}= (1 2 3 4 … 26), that is,

$$\begin{bmatrix} 1 & 2 & 3 & \ldots & 25 & 26 \\ 2 & 3 & 4 & \ldots & 26 & 1 \end{bmatrix}$$

Firstly white space are removed from X, that is, X=MEETATONE. Then apply the permutation on message X. The resultant cipher text will be NFFUBUPOF. To recover the message X, inverse permutation has to be applied. This is referred as K^{-1}.

$$\begin{bmatrix} 1 & 2 & 3 \ldots. & 26 \\ 26 & 1 & 2 \ldots. & 1 \end{bmatrix}$$

Bob can pull out the information with d_{Π}= (1 26 25 … 3 2).■

Cryptography is of two types, public key cryptography and secret (private, symmetric) key cryptography. Private-key cryptography is what we have seen so far. There are two participants: a sender and a receiver. They choose a private key K for their secure communication. In public-key cryptography secret key K has two components $\langle \text{public} \mid \text{private} \rangle$. Public part of the secret key is known to everybody but private part is only known to the receiver. Now when the sender wants to send some information to the receiver over an insecure channel, he encrypts the data with the receiver's public key. Only the receiver can decrypt with his private key while no other can.

1.3 Secret Sharing Cryptosystem

In private-key cryptography, giving a secret key K to an individual may not be secure. That individual may misuse the secret key K. For example a car company, such as GMC or Ford, has developed its future sports car model. This new model needs to be saved in a secure environment so that competitive companies would not know anything about the new model. Here security of such a project may not depend on a single person. Because the individual may misuse the information that he/she has, hence the idea of a secret sharing scheme was introduced in cryptography. Secret sharing scheme is a method through which a secret key K is shared among the group of participants. That is, each participant receives a share or shares about the secret key K. To determine the secret key K, all participants belonging to the authorized group must pool their share together. If one or more participants from the authorized group go missing, then the remaining cannot determine the secret key K. The main motivation for secret sharing scheme is to safeguard the secret information by protecting the key, as key can't be trusted in the

hands of one. One solution is to distribute it, such that the access is no longer in the hands of one.

Example 1.3.1 Consider two people who share a business and a joint checking account related to it. For withdrawal a check needs to be signed by both. If any one of the signatures goes missing on the check, they won't be able to withdraw the amount from their account. So without the knowledge of both, withdrawal of an amount is impossible. In this case shares are the signatures of individuals. The secret key (check) is recognized only when both signatures come together.■

In cryptography different secret sharing schemes exist. Secret Sharing Scheme was first proposed by Adi Shamir and George R. Barkley in 1979 [9, 13]. Even the very first secret sharing scheme was named after Adi Shamri as "Shamir threshold scheme". Later on, different schemes came into existence. Some of them are Monotone circuit construction [13], Brickell vector space construction [13] and many more.

1.4 Some Basic Definitions

In this subsection, we will cover several definitions which are used throughout the thesis.

Participants: Participants are a group of people, who participate in key distribution. Commonly denoted as $\mathcal{P} = \{P_1, P_2, \dots, P_n\}$.

Dealer: $\mathcal{D}$ is a special participant where $\mathcal{D} \notin \mathcal{P}$. He is responsible for share distribution to .

Authorized set: A subset B of $\mathcal{P}$.When all the participants in B pool their share; they should be able to determine the secret K.

Example 1.4.1 $B = \{P_1,\ P_2\}$, means P_1, P_2 together are authorized to pull out key K.■

Access Structure: Set of authorized groups Γ is called as access structure. That is $\Gamma = \{B_i | B_i \text{ is an authorized set}\}$.

Secret sharing scheme: Secret sharing scheme is an algorithm through which shares of secret key K are distributed to corresponding participants in such a way, that only the set of authorized participants can recover the secret key K while no other can.

CHAPTER 2: SCHEMES

In this chapter, we will summarize three well-known secret sharing schemes. They are Shamir's threshold scheme, monotone circuit construction, and Brickel's vector space construction. Since all these three secret sharing schemes are perfect secret sharing schemes, we first define perfect secret sharing scheme.

Definition 2.1: A perfect secret sharing scheme is a method or algorithm through which the secret key K is distributed among a set of n participants $\mathcal{P}$ in such a way that the following two properties are satisfied:

- Every authorized set $B \in \Gamma$ can determine the secret key K, if their shares are pooled together.
- Every unauthorized set $\notin \Gamma$ cannot determine secret key K, even if their shares are pooled together.

Example 2.1: Consider $\mathcal{P} = \{P_1, P_2, P_3, P_4\}$ be the set of participants. Let the access structure $\Gamma = \{\{P_1, P_2\}, \{P_2, P_3\}, \{P_3, P_4\}\}$. Based on Γ the authorized sets B_i are $B_1 = \{P_1, P_2\}$, $B_2 = \{P_2, P_3\}$, $B_3 = \{P_3, P_4\}$. The shares of the secret key K are distributed to participants $\mathcal{P}$ based on the access structure Γ which satisfies the definition 2.1. If the authorized group B is the subset of a set say C, then C will automatically become authorized as B. This has been illustrated in Figure 2 where the bold rectangular boxes are authorized and circular ones are unauthorized.

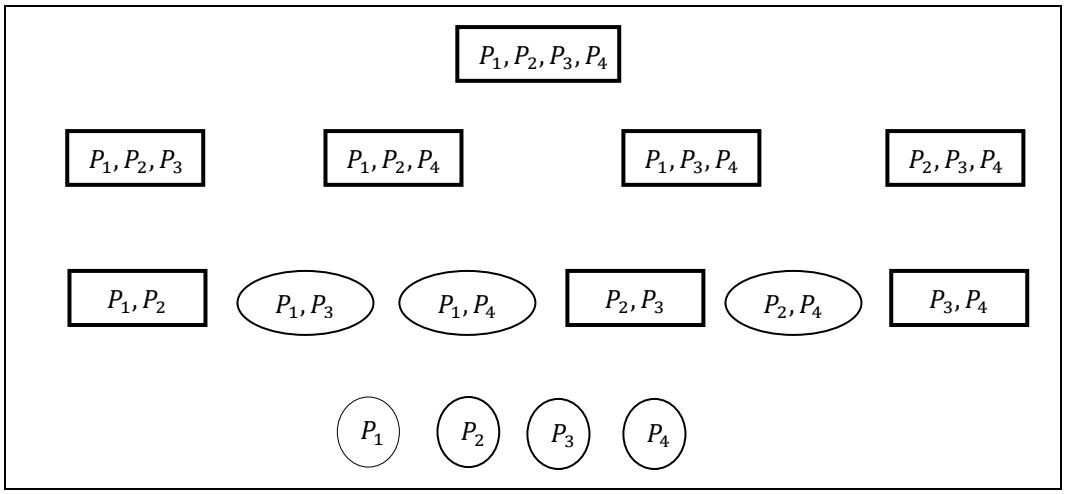

Figure- 2. Illustration of authorized sets.

Depending upon the above access structure Γ, any group say $\notin \Gamma$ cannot determine the secret key K. For Example, $= \{P_1, P_3\}$ is an unauthorized set and cannot determine the secret key K.■

2.1 The Shamir Threshold Scheme

Shamir Threshold Scheme is ideal for safeguarding the information, by allowing only authorized participants to reveal the information. In a real time situation according to Time Magazine [15], the Russian President, defense minister and defense ministry tags by carrying a briefcase of electronic control for nuclear ignition which is called "Foot Ball" by the Americans. So out of these three, any two can conjunct to ignite the nuclear weapon.

Definition 2.1.1 [13]: Let t be an integer with $t \leq n$. (t, n) is a threshold scheme by which secret key K is shared among the set of n participants, such that t or more participants can compute the secret key K. But not less than t participants can compute the secret key K.

Threshold access structure is $\Gamma = \{B \subseteq \mathcal{P} : |B| \geq t\}$.

Example 2.1.1: $\Gamma = \{B \subseteq \mathcal{P} : |B| \geq 3\}$ where $\mathcal{P} = \{P_1, P_2, \dots, P_n\}$. Set of any participants less than 3 cannot recover the secret key K.

Algorithm 2.1 [7,13]*: General (t, n) Shamir threshold*

Input: Let $\{P_1, P_2, \dots, P_n\}$ be a set of participants and let t be an integer value such that $t \leq n$.

Output: Shares y_i for each corresponding participants P_i, where $1 \leq i \leq n$.

Initial process:

1. /* $\mathcal{D}$ chooses distinct x values, such that $x \neq 0$ and $x \in Z_p$, where p is a prime number. */
   $for\ i := 1\ to\ n$
   $P_i := x_i$;
   //The shares x_i are distributed to participants P_i as public shares.

Share Distribution:

1. // $\mathcal{D}$ secretly chooses a secret key value K, where $K \in Z_p$.

2. $for\ i := 1\ to\ t-1$

   $a_i := random(\,)\ mod\ p, where\ a_i\ are\ distinct.$

3. // $\mathcal{D}$ constructs polynomial $a(x)$
   Consider $v := 0$;
   $for\ j := 1\ to\ t-1$
   $v := v + a_j x^j$;

$a(x) := K + v \ mod \ p;$

4. // $\mathcal{D}$ computes the shares for each corresponding participant P_i.

   $for \ i := 1 \ to \ n$

   $y_i = a(x_i);$

   $P_i := y_i;$

   //These shares y_i distributed to corresponding participant P_i, are private.

Example 2.1.2: Let a set of participants $\mathcal{P} = \{P_1, P_2, \ldots, P_5\}$ and let a threshold be (3, 5), that is, $t = 3$ and $n = 5$.

Step 1: Let the access structure $\Gamma = \{ B \mid B \subseteq \mathcal{P}, \ |B| \geq 3\}$.

Step 2: $\mathcal{D}$ selects elements $\{x_i = i \mid 1 \leq i \leq 5\}$ as public share and distributes to set of participants $\mathcal{P}$.

Step 3: $\mathcal{D}$ selects secret key $K = a_0 = 7$ and selects $t - 1$ random elements a_i as

$a_1 = 5 \ and \ a_2 = 1.$

Step 4: $\mathcal{D}$ Constructs polynomial $a(x) = a_0 + a_1 x + a_2 x^2$

$$a(x) = x^2 + 5x + 7.$$

Step 5: Computation of shares by $\mathcal{D}$:

$$y_1 = a(1) = 13 \ mod \ 17$$

$$= 13$$

$$y_2 = a(2) = 21 \ mod \ 17$$

$$= 4$$

$$y_3 = a(3) = 31 \; mod \; 17$$

$$= 14$$

$$y_4 = a(4) = 43 \; mod \; 17$$

$$= 9$$

$$y_5 = a(5) = 57 \; mod \; 17$$

$$= 6$$

Shares generated for each participant is as follows: P_i *{*x_i *(publicly known),* y_i *(private)}*

P_1 : $\{1, 13\}$

P_2 : $\{2, 4\}$

P_3 : $\{3, 14\}$

P_4 : $\{4, 9\}$

P_5 : $\{5, 6\}$. ■

Key Recovery:

Key can be recovered by computing $a(x)$ by authorized group B_i.

$$a(x) = K = \sum_{j=1}^{t} y_{i\,j} \prod_{1 \le K \le t, K \ne j} \frac{x - x_{ik}}{x_{ij} - x_{ik}}$$

Example 2.1.2: From the above example, let's compute the K value from the available secret shares y_i distributed to authorized group $B = \{P_2, P_3, P_4\}$.

$$a(x) = \frac{(x-3)(x-4)}{(2-3)(2-4)} \cdot 4 + \frac{(x-2)(x-4)}{(3-2)(3-4)} \cdot 14 + \frac{(x-2)(x-3)}{(4-2)(4-3)} \cdot 9$$

$$a(0) \;=\; -61 \; mod \; 17 \;=\; -10 \; (-10 \; in \; mod \; 17 \; is \; 7) = \; 7$$

$\therefore \; K = 7$. Hence secret key K is recovered. ■

Example 2.1.3: From the above example 2.1.1, let's compute the K value from the available secret shares y_i by unauthorized group $= \{P_2, P_3\}$.

$$a(x) \;=\; \frac{(x-3)}{(2-3)} \,.\, 4 \;+\; \frac{(x-2)}{(3-2)} \,.\, 14$$

$$a(0) \;=\; -16 \; mod \; 17 \;= 1$$

$\therefore \; K \neq 1$. Hence secret key K is not recovered; similarly any t-1 participants forming a group cannot recover the secret key K . ■

Disadvantage of Shamir threshold (t, n) scheme

This scheme is said to be theoretically insecure, as t or more participants from n can recover the secret key K. Thus, there is no way to choose participants from $\mathcal{P}$ to form a group of dealer $\mathcal{D}$ choice.

2.2 Monotone Circuit Construction Scheme

Monotonic circuit construction is attributed to Benaloh [13] and Leichter [13]. This construction consists of a Boolean circuit with "OR" and "AND" gates represented in disjunctive normal form.

Definition 2.2.1: A circuit C with n inputs and m outputs, constructed with AND, OR and NOT gates is called as boolean circuit C.

Definition 2.2.2 [9]**:** Let $C(x_1, x_2, x_3, \dots, x_n)$ be a Boolean circuit. If C can be written as $C(x_1, x_2, x_3, \dots, x_n) = \bigvee_{s=1\ to\ k}(\bigwedge x_{ij})$, then C is said to be in disjunctive normal form (DNF).

Definition 2.2.3: A Boolean circuit in DNF form with n inputs and 1 output, constructed with only OR and AND gates is called monotone circuit.

Example 2.2.1: Boolean circuit $C(x_1, x_2, x_3) = (x_1 \wedge x_2) \vee (x_2 \wedge x_3)$ in DNF form is a monotone circuit.■

Before describing the algorithm to construct access structure from given monotone circuit, we will demonstrate this construction for the above circuit.

Example 2.2.2: Let monotone circuit $C(x_1, x_2, x_3) = (x_1 \wedge x_2) \vee (x_2 \wedge x_3) = m_1 \vee m_2$, for $m_1 = x_1 \wedge x_2$ *and* $m_2 = x_2 \wedge x_3$. First let us determine a set of participants $\mathcal{P} = \{P_1, P_2, P_3\}$, where each P_i is corresponding input x_i in C. Next we determine authorized sets $B_1 = \{P_1, P_2\}$ since x_1, x_2 are in m_1 and $B_2 = \{P_2, P_3\}$ since x_2, x_3 are in m_2. Now here is the access structure $\Gamma_c = \{B_1, B_2\}$.

The following algorithm gives the construction of Γ_C for any given monotone circuit C**.**

Algorithm 2.2.1**:** *Construction of access structure* Γ_c.

Input: Monotone boolean circuit $C(x_1, x_2, x_3, \dots, x_n) = m_1 \vee m_2 \dots \vee m_k$.

Output: Access structure Γ_c.

Construction of Γ_C*:*

1. $for\ i := 1\ to\ n$

   $P_i \leftrightarrow x_i$

   $\mathcal{P} := \{P_1, P_2, \dots, P_n\}$

2. $for\ i := 1\ to\ k$

   $B_i := \{P_j \mid x_j \in m_i\}$.

3. $\Gamma_C := \{B_1, B_2, \dots, B_k\}$.

The following is the secret sharing scheme (algorithm) realizing access structure Γ_C.

Algorithm 2.2.2 [11]**:** *Monotone Circuit Construction* (C)

Input: Monotone boolean circuit C

Output: shares $\{y_1, y_2, \dots, y_n\}$ to participants $\{P_1, P_2, \dots, P_n\}$.

Initial process:

1. We construct access structure Γ_C by call algorithm 2.2.1.

2. Assign all gates as $G_i\ for\ 1 \leq i \leq k$.

*Share Distribution***:**

1. // k Inputs and 1 output wire of OR ($\vee$) gate is assigned with K.

2. $for\ i \coloneqq 1\ to\ k$

   Let $Gate\ G_i\ with\ t\ inputs$

   2.1 $\mathcal{D}$ choose $t-1$ variables from Z_p, say $a_1, a_2, \dots, a_{t-1}$.

   2.2 $\mathcal{D}$ assign first $t-1$ inputs of G_i with $a_1, a_2, \dots, a_{t-1}$.

   2.3 $\mathcal{D}$ assign last input wire of G_i with $K - (\ a_1, a_2, \dots, a_{t-1})\ mod\ p$.

Example 2.2.3: Let $\mathrm{C}(x_1, x_2, x_3, x_4, x_5, x_6) = m_1 \vee m_2 \vee m_3$ be monotone boolean circuit for $m_1 = x_1 \wedge x_2 \wedge x_3,\ m_2 = x_1 \wedge x_3 \wedge x_4\ and\ m_3 = x_5 \wedge x_6$.

Step 1: Where $m_1,\ m_2\ and\ m_3$, are sent to **Algorithm 2.2.1** as an input.

Step 2: Each x_i is corresponded to each participant P_i, thus forming a set of participants $\mathcal{P} = \{P_1, P_2, \dots, P_6\}$.

Step 3: Where $B_1 = \{P_1, P_2, P_3\}, B_2 = \{P_1, P_3, P_4\}\ and\ B_3 = \{P_5, P_6\}$, as B is a set of participants belonging to m_k.

Step 4: $\Gamma_C = \{B_1, B_2, B_3\}$, such that $\Gamma_C = \big\{\{P_1, P_2, P_3\}, \{P_1, P_3, P_4\}, \{P_5, P_6\}\big\}$.

Step 5: Γ_C and monotonic circuit C are sent as an input to **Algorithm 2.2.2** to compute and distribute the shares.

Step 6: Let's suppose G_1, G_2, G_3 be AND gates and G_4 be OR gate.

Step 7: All inputs and output to G_4 be assigned with K.

Step 8: For gate G_1 with inputs $t = 3$. $\mathcal{D}$ chooses $t - 1$ elements from Z_p as a_1 and a_2.

The last input wire is assigned with $K - a_1 - a_2 \; mod \; p$.

Step 9: Step 8 is repeated until all inputs to gates G_i are assigned with values, as follows.

Step 10: For G_2: $b_1,\ b_2, K - b_1 - b_2$ and G_3: $c_1, K - c_1$.

Step 11: So therefore each participant receives the following as shares which are private in nature.

P_1: $\{a_1, b_1\}$

P_2: $\{K - a_1 - a_2\}$

P_3: $\{a_2, b_2\}$

P_4: $\{K - b_1 - b_2\}$

P_5: $\{c_1\}$

P_6: $\{\ K - c_1\}$

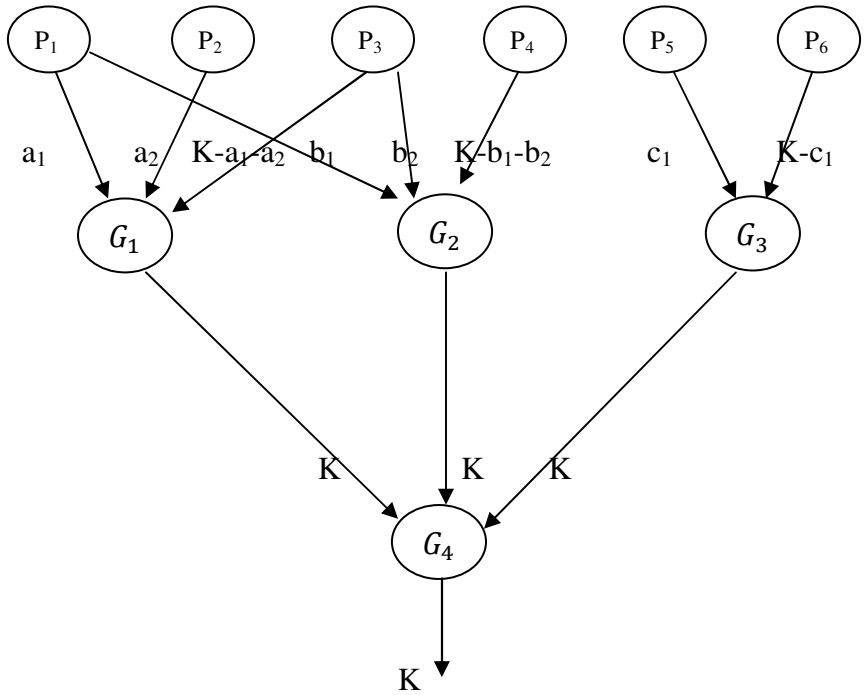

Figure- 3. Monotone shares distribution to set of participants

Secret Key Recovery:

1. For any B_i authorized set, secret key K can be recovered by summing up their shares to $mod\ p$.

Example 2.2.4: Illustration of secret key K recovery by authorized party

Consider $B = \{P_1, P_2, P_3\}$ from above example 2.2.3. When the shares of participants in B are pooled together, it gives out the key as shown below. Similarly any other authorized groups of Γ can compute the secret key K.

$$a_1 + a_2 + (K - a_1 - a_2)\ mod\ p = K. \blacksquare$$

Example 2.2.5: Illustration of secret key K recovery trail by unauthorized party

Consider $= \{P_1, P_2, P_4\}$ from above example 2.2.2. When the shares of participants in are pooled together, it does not give out the key as shown below. Similarly any other unauthorized groups $\notin \Gamma$ cannot compute the secret key K.

$$a_1 + a_2 + (K - b_1 - b_2) \ mod\ p \ \neq K. \blacksquare$$

2.3 Brickell Vector Space Constructions

The vector space construction [4] is introduced by Brickell. This generalization by Bertilsson [6], leads to secret sharing scheme in which the secret can be computed efficiently by each qualified group.

Let's suppose,

d be a positive number, where $d \geq 2$.

Z_p is set of positive numbers, where p is prime number.

$(Z_p)^d$ denote the vector space over Z_p .

Γ be defined as the access structure.

Let's assume a function exists : $\mathcal{P} \rightarrow (Z_p)^d$ such that it satisfies the following property

$(\mathbf{1}, \mathbf{0}, \mathbf{0}, \ldots, \mathbf{0}) \in \langle\, (\boldsymbol{P_i}) \mid \boldsymbol{P_i} \in \boldsymbol{B} \,\rangle \Leftrightarrow \boldsymbol{B} \in \boldsymbol{\Gamma}$ **(2.3)**,

The construction of function is a matter of trial and error. Though we see some construction generated randomly by trial method satisfying property 2.3 in example 2.3.1.

In an upcoming chapter we will discuss a proposed algorithm to construct the function , satisfying only some access structure.

Algorithm 2.3[5,11]***:*** *Brickell vector space share distribution*

Input: Function satisfying property 2.3, access structure Γ and set of participant $\mathcal{P}$

Output: share $\{y_1, y_2, \dots, y_n\}$ to participants $\{P_1, P_2, \dots, P_n\}$.

Initial process:

1. $for\ i \coloneqq 1\ to\ n$

   $P_i \coloneqq (P_i),\ where\ (P_i) \in (Z_p)^d$.

   // $\mathcal{D}$ distributes $(P_i)\ to\ P_i$ .These shares are public in nature.

Share Distribution:

1. $\mathcal{D}$ chooses secret K form Z_p.

2. $\mathcal{D}$ selects $d-1$ secret elements $a_2, a_3, \dots, a_d$ from Z_p .

3. $\mathcal{D}$ construct the vector $\bar{a} = (K, a_2, a_3, \dots, a_d)$.

4. // $\mathcal{D}$ computes shares y_i corresponding to P_i.

   $for\ i \coloneqq 1\ to\ n$

   $y_i \coloneqq \bar{a}.(P_i);$

Example2.3.1: Let a set of participants $\mathcal{P} = \{P_1, \dots, P_4\}$, $\Gamma = \{\{P_1, P_2, P_3\}, \{P_1, P_4\}\}$ be access structure.

Step 1: By trial and error we can find the following function satisfying 2.3.

$$(P_1) = (0,1,0)$$

$$(P_2) = (1,0,1)$$

$$(P_3) = (0,1,-1)$$

$$(P_4) = (1,1,0)$$

Step 2: Based on the function $d = 3$.

Step 4: First we need to show that this function satisfies the property 2.3.

For that we need to find the coefficient c_i for function (P_i) .

$$(1,0,0) = \sum c_i\,(P_i).$$

From access structure Γ, $B_1 = \{P_1, P_4\}$ and $B_2 = \{P_1, P_2, P_3\}$.

For B_1 :

$$(1,0,0) = c_1\,(P_1) + c_4\,(P_4)$$

$$= c_1(0,1,0) + c_4\,(1,1,0)$$

$$= (c_4, c_1 + c_4, 0).$$

$$\therefore \quad c_4 = 1 \; and \; c_1 = -1.$$

For B_2 :

$$(1,0,0) = c_1\,(P_1) + c_2\,(P_2) + c_3\,(P_3)$$

$$= c_1(0,1,0) + c_2\ (1,0,1) + c_3\ (0,1,-1)$$

$$= (c_2, c_1 + c_3, c_2 - c_3).$$

$$\therefore \quad c_1 = -1, c_2 = 1\ and\ c_3 = 1.$$

Now when we substitute the c_i in $\sum c_i\ (P_i)$, it must generate (1,0,0).

So, we have

$$-(P_1) + (P_2) + (P_3) = -(0,1,0) + (1,0,1) + (0,1,-1) = (1,0,0).$$

$$-(P_1) + (P_4) = -(0,1,0) + (1,1,0) = (1,0,0).$$

Thus we can say that this function satisfies the property 2.3.

Step 5: $\mathcal{D}$ chooses secret key $K = 8$.

Step 6: $\mathcal{D}$ selects $d - 1$ secret elements $a_2 = 4\ and\ a_3 = 1$.

Step 7: $\mathcal{D}$ constructs vector $\bar{a} = (8, 4, 1)$.

Step 8: $\mathcal{D}$ computes shares $y_i = \bar{a}.(P_i)$.

$$y_1 = (8, 4, 1).(0, 1, 0)$$

$$= 4\ mod\ 17$$

$$= 4$$

$$y_2 = (8, 4, 1).(1, 0, 1)$$

$$= 4\ mod\ 17$$

$$= 9$$

$$y_3 = (8,4,1).(0,1,-1)$$

$$= 3 \; mod \; 17$$

$$= 3$$

$$y_4 = (8,4,1).(1,1,0)$$

$$= 12 \; mod \; 17$$

$$= 12$$

Step 5: Each participant receives secret shares as follows.

P_1: 4

P_2: 9

P_3: 3

P_4: 12.■

Secret key Recovery

1. Assume $\bar{a}.(1,0,\ldots 0) = K$, such that

$$K = \bar{a} . \sum c_i \; (P_i) \; where \; i: P_i \in B.$$

That is $K = \sum c_i \; . y_i \; , where \; i: P_i \in B.$

Example 2.3.2: Based on the above example, let's take $B_1 = \{ P_1, P_2 , P_3 \}$ as an authorized set.

Step 1: Pooling all shares of participants in B_1 together.

$$(1,0,0).(K, a_2, a_3) = c_1 . y_1 + c_2 . y_2 + c_2 . y_3$$

$$(1,0,0).(K,4,1) \;=\; -4+9+3$$

$$K \;=\; 8 \; mod \; 17$$

$$K \;=\; 8.$$

∴ Hence secret key K is recovered.■

Example 2.3.3: Based on above example, let's consider an unauthorized set $= \{P_1, P_2\}$.

Step 1: The unauthorized set will not be able to find the coefficient c_i, as polynomial contradiction would occur. Let's try to find c_i, for defined unauthorized set .

Step 2: $(1,0,0) = c_1(0,1,0) + c_2\,(1,0,1)$

$$= (c_2, c_1, c_2)$$

$$\therefore \quad c_1 = 0, c_2 = 1 \; or \; 0 \;\; .$$

Thus a contradiction for c_2 , as it has two solutions. Similarly, any unauthorized group cannot recover the secret key K.

Step 3: Without the coefficient, the unauthorized group cannot compute secret key.■

CHAPTER 3: CONSTRUCTING FUNCTION FOR SPECIAL ACCESS STRUCTURE

In this chapter, we describe the construction of the function in polynomial time for some special access structure. First we will give some definitions from graph theory.

Definition 3.0[12]**:** A graph $G(V, E)$ is called a bipartite graph if its vertex set V can be partitioned into two disjoint subsets A and B such that every edge in G joins a vertex in A to a vertex B.

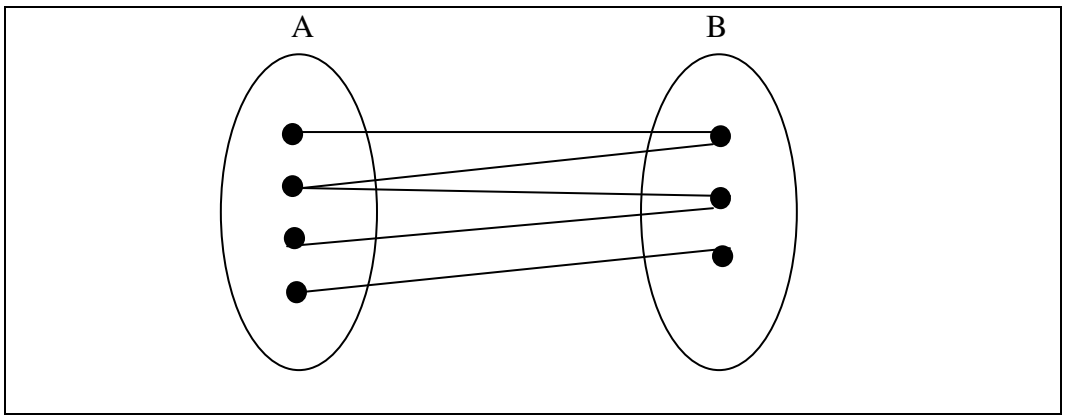

Figure- 4.Bipartite Graph G ■

Definition 3.1[12]**:** A bipartite graph $G(V, E)$ in which every vertex of A is adjacent to every vertex in B is called a complete bipartite graph. Where A and B are partitioned subsets of the vertex V of G. If |A|=m and |B|=n, then the complete bipartite graph is denoted by $K_{m,n}$ and has mn edges.

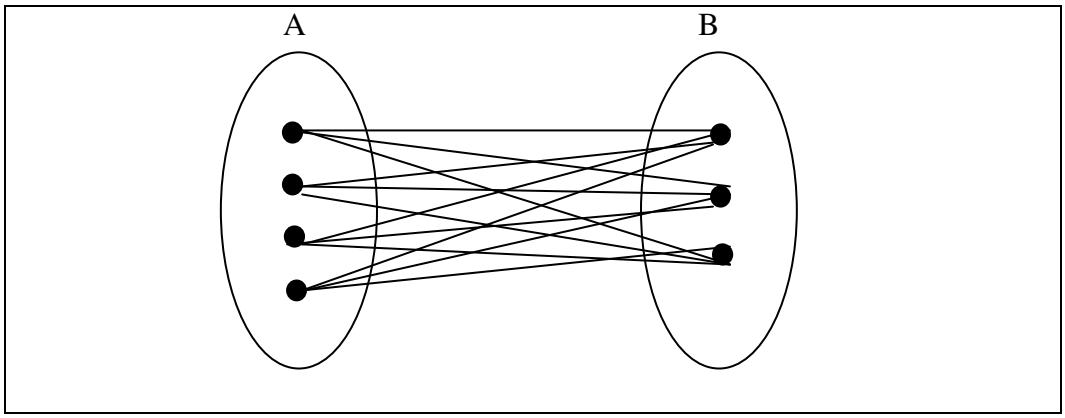

Figure- 5.Complete Bipartite Graph $G: K_{4,3}$ ■

Definition 3.2: A graph $G(V, E)$ is called multipartite graph if its vertex set V can be partitioned into k disjoints subsets say $V_1, V_2, \dots, V_k$, such that every edge in G joins vertex from V_i and V_j, where $i \neq j$.

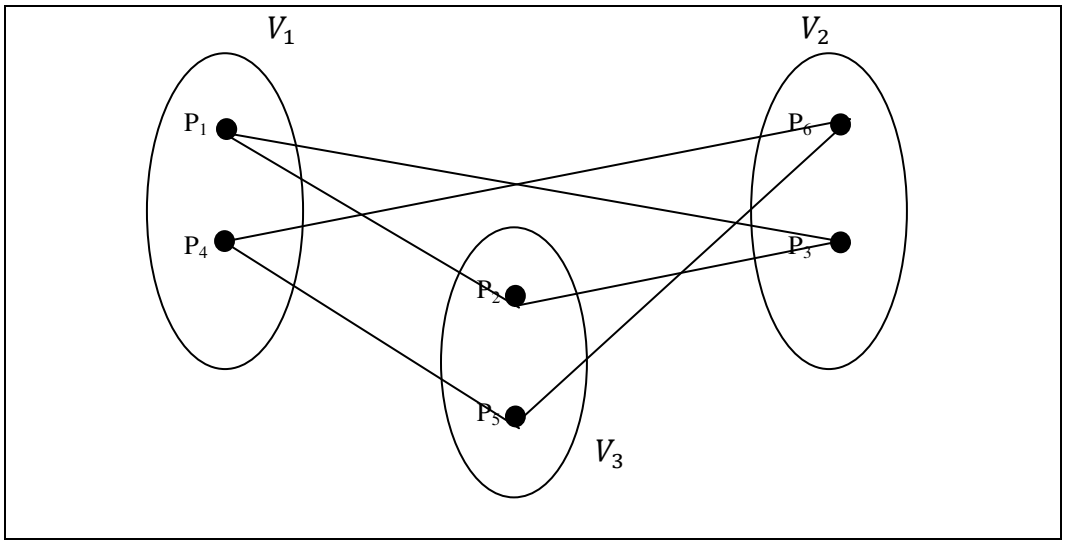

Figure- 6. Multipartite Graph G■

Definition 3.3[5]**:** Multipartite graph is called complete multipartite graph if all the vertices V_i is connected to all the vertices in V_j for all $i \neq j$. The complete multipartite graph is denoted by $K_{n_1,n_2,n_3,\dots,n_l}$ if $|V_i| = n_i, 1 \leq i \leq l$.

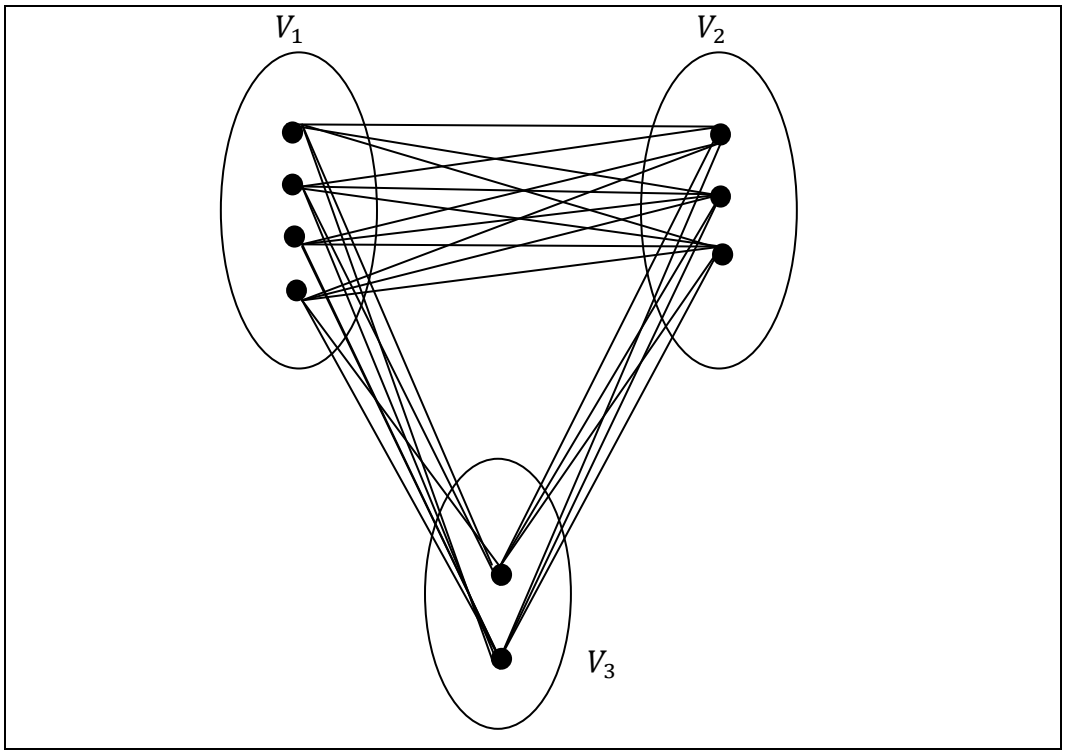

Figure- 7.Complete Multipartite Graph $G: K_{2,3,4}$ ■

3.1 Construction of function for special access structure I

Access structure is defined as follows: Let $G(V, E)$ be a complete multipartite graph, then we can construct access structure Γ_E as follows:

$$\mathcal{P} = V, \Gamma_E = \{set\ of\ edges\ of\ G\} = E$$

Example 3.1.1: Let G be a complete bi-partite graph given in Figure 7. Then the set of participants $\mathcal{P} = V = \{P_1, P_2, P_3, P_4, P_5, P_6\}$ and access structure Γ_E is

$$\{\{P_1, P_3\}, \{P_1, P_4\}, \{P_1, P_6\}\{P_2, P_3\}, \{P_2, P_4\}, \{P_2, P_6\}, \{P_5, P_3\}, \{P_5, P_4\}, \{P_5, P_6\}\}$$

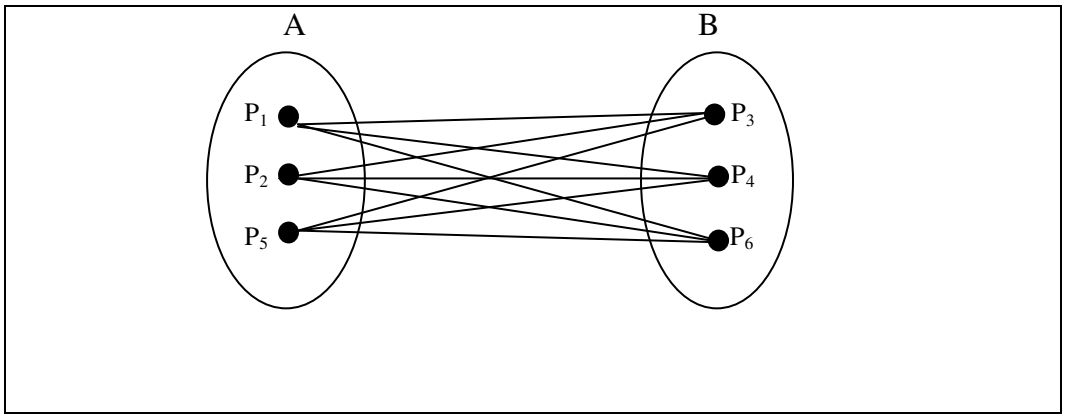

Figure- 8.Complete Multipartite Graph $G: K_{3,3}$ ■

Theorem 1.0 [11]**:** Suppose $G = (V, E)$ is a complete multipartite graph. Then there is an ideal scheme realizing the access structure Γ_E on participant set V.

Proof: Let $V = V_1 \cup V_2 \cup \ldots \cup V_l$ be a set of vertices of G. Let $x_1, x_2, \ldots, x_l$ be distinct elements of Z_p, where $p \geq l$. Let $d = 2$. Variable x_i is representing set V_i for *i=1,2,…,l*

For every participant $v \in V_i$, define a function $(v) = (x_i, 1)$. Now we can show that this function satisfies (2.3). Let suppose, $B = \{P_i, P_j\} \in \Gamma$ such that $P_i \in V_i \; and \; P_j \in V_j$.

Thus $(P_i) = (x_i, 1)$ and $(P_j) = (x_j, 1)$, where $x_i \neq x_j$. Hence

$$a_1.(P_i) + a_2.(P_j) = (1,0)$$

$$a_1.(x_i, 1) + a_2.(x_j, 1) = (1,0)$$

$$a_1 + a_2 = 0 \; and \; a_1.x_i + a_2.x_j = 1$$

$$a_2 = -a_1, \;\; a_1.x_i - a_1.x_j = 1$$

$$a_1(x_i - x_j) = 1$$

as $x_i - x_j \neq 0, an\ inverse\ exist\ such\ that\ a_1 = (x_i - x_j)^{-1}$. Therefore shares of any authorized set can generate (1, 0). Let suppose, $C = \{P_{i_1}, P_{i_2}, \dots, P_{i_l}\}$ is an unauthorized set. By the definition of complete multipartite graph, there exist i such that $C \subseteq V_i$ .Thus $(P_{i_1}) = (P_{i_2}) = (P_{i_3}) = \cdots (P_{i_l}) = (x_i, 1)$. Hence

$$\sum a_i \left(P_{i_l}\right) = (1,0)$$

$$a_1.\left(P_{i_1}\right) + \cdots + a_l.\left(P_{i_l}\right) = (1,0)$$

$$a_1.(x_i, 1) + \cdots + a_l.(x_i, 1) = (1,0)$$

$$a_1 + a_2 + \cdots + a_l = 0\ and\ a_1.x_i + \dots + a_l.x_i = 1$$

$$x_i(a_1 + a_2 + \cdots + a_l) = 1$$

$$(a_1 + a_2 + \cdots + a_l) = {x_i}^{-1}$$

Thus, it is a contradiction.

***Algorithm 3.1.1*:** *Construction of from complete multipartite graph* $G(V, E)$.

Input: Complete multipartite graph for $K_{n_1,n_2,n_3,\dots,n_l}$

Output: $:\mathcal{P} \rightarrow {Z_p}^2$

Process:

1. Determine vertex $V_i\ for\ i = 1,2,..l$.

2. Find l distinct elements $x_1, x_2, \ldots, x_l$ of $Z_p, where\ p > l$ and x_i is associated with V_i.

3. $for\ i = 1\ to\ |V|$

   Compute (P_{j_i})=($x_i, 1$), where $P_{j_i} \in V_i$

4. Construct as follows

$$= \begin{bmatrix} x_1 & 1 \\ x_2 & 1 \\ \ldots & \ldots \\ x_l & 1 \end{bmatrix}$$

5. Return .

Example 3.1.2: Illustration of the above **Algorithm 3.1.1**

Step 1: Let's derive complete multipartite graph G from Figure 7 as input.

Step 2: Vertex $V = \{V_1, V_2\} = \{\{P_1, P_2, P_5\}, \{P_3, P_4, P_6\}\}$.

Step 3 Let distinct elements be $\{x_1, x_2\}$ of Z_p such that $x_1 \leftrightarrow V_1\ and\ x_2 \leftrightarrow V_2$.

Step 4: For every $P_{i_j} \in V_i$ we compute $\left(P_{i_j}\right) = (x_i, 1)$.

Step 5: That is for V_1: $(P_1) = (x_1, 1), (P_2) = (x_1, 1)\ and\ (P_5) = (x_1, 1)$.

Step 6: Similarly for V_2: $(P_3) = (x_2, 1), (P_4) = (x_2, 1)\ and\ (P_6) = (x_2, 1)$.

Step 7: Matrix constructed as follows:

$$= \begin{bmatrix} x_1 & 1 \\ x_1 & 1 \\ x_2 & 1 \\ x_2 & 1 \\ x_1 & 1 \\ x_2 & 1 \end{bmatrix}$$

■

3.2 Construction of function for special access structure II

Access structure in this section is represented with a multipartite graph G. But the graph G is not a complete multipartite graph. Here is the definition of such access structure.

Let $\mathcal{P} = \{P_1, P_2, P_3, \dots, P_n\}$ be set of participants and $\Gamma = \{B_1, B_2, \dots, B_k\}$ be an access structure with the following two properties:

- $B_i \cap B_j = \emptyset \; for \; all \; i \neq j.$
- $|B_j| = m \; for \; j = 1,2,3,..k.$

Example 3.2.1: Let G (V, E) be a multipartite graph with vertex set V $= \{P_1, P_2, P_3, P_4, P_5, P_6\}$ depicted in Figure 8.

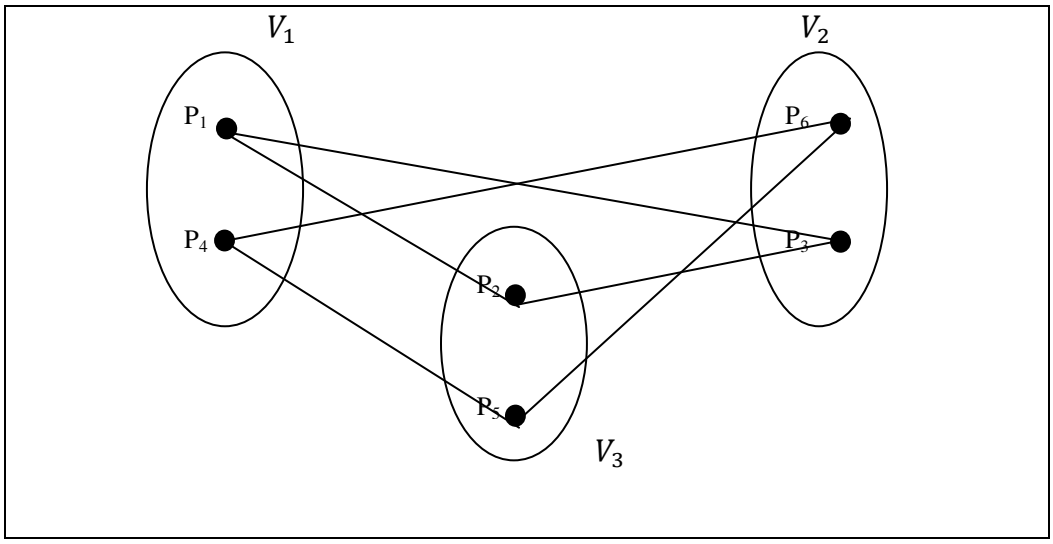

Figure- 9. Multipartite graph representing Γ

From the above multipartite graph, we construct access structure as follows:

$\Gamma = \{B_1, B_2\} = \{\{P_1, P_2, P_3\}, \{P_4, P_5, P_6\}\}$, where a set of participants are a vertex set of graph G.

The following is the algorithm to construct a function satisfying (2.3) for a given access structure Γ. ■

***Algorithm 3.2*:** *Construction of matrix M representing function*

Input: Let $\mathcal{P} = \{P_1, P_2, P_3, \dots, P_n\}$ be set of participants and

let $\Gamma = \{B_1, B_2, \dots, B_k\}$ be access structure such that $B_i \cap B_j = \emptyset$ for all $i \neq j$ and $|B_i| = m$ for $i = 1,2,3, \dots, k$.

Output: Matrix M representation of function .

Initial process:

1. Calculate column$:= 2 * m - 1$; // no of columns for matrix M.

2. $for\ s := 1\ to\ k$

   Create $m \times 2m - 1$ matrix M_s with zero entries.

3. Find k distinct elements of Z_p such that $x_k > x_{k-1} > \cdots > x_1 > 1$.

Construction:

1. $for\ s := 1\ to\ k$

   $//\ update\ m \times 2m - 1\ matrix\ M_s$

2. $for\ i := 1\ to\ m$

   $(M_s)_{i,i} = 1$

3. $for\ i := 1\ to\ m - 1$

   $(M_s)_{i,i+1} = 1$

4. $for\ i := 1\ to\ m - 1$

   $(M_s)_{i,m+i} = x_s$

   $(M_s)_{i+1,m+i} = x_s$

5. $M = \begin{bmatrix} M_1 \\ M_2 \\ \dots \\ M_s \end{bmatrix}$

6. Return M.

Example 3.2.4:

Let $\mathcal{P} = \{P_1, P_2, P_3, \dots, P_6\}$ be a set of participants and access structure $\Gamma = \{B_1, B_2\} = \{\{P_1, P_2, P_3\}, \{P_4, P_5, P_6\}\}$. Γ satisfies both properties given above. $|B_1| = |B_2| = m = 3$ and $k = 2$. Let us take $p = 5$ be a prime number.

Step 1: Construct two 3×5 matrices M_1, M_2 with 0 entries

Step 2: Choose two distinct elements $x_1 = 2, x_2 = 3$ of Z_5

Step 3: For M_1: $M_{i,i} = 1 \;\; for\ i \coloneqq 1\ to\ 3$.

1	0	0	0	0
0	1	0	0	0
0	0	1	0	0

Step 4: $M_{i,i+1} = 1\ for\ i \coloneqq 1\ to\ 2$.

1	1	0	0	0
0	1	1	0	0
0	0	1	0	0

Step 5: Consider $M_{i,3+i} = 2$ and $M_{i+1,3+i} = 2\ for\ i \coloneqq 1\ to\ 2$

1	1	0	2	0
0	1	1	2	2
0	0	1	0	2

Step 6: Repeat the above steps with $x_2 = 3$ for M_2.

1	1	0	3	0
0	1	1	3	3
0	0	1	0	3

Step 7: $M = \begin{bmatrix} M_1 \\ M_2 \end{bmatrix}$

Step 8: resultant M

1	1	0	2	0
0	1	1	2	2
0	0	1	0	2
1	1	0	3	0
0	1	1	3	3
0	0	1	0	3

Let's prove that the above matrix M 3.1 satisfies the property 2.3. First if we take the first authorized set B_1, then

$(1,0,0,0,0) = (1,1,0,2,0) - (0,1,1,2,2) + (0,0,1,0,2)$.

Next take the other authorized set B_2, then

$(1,0,0,0,0) = (1,1,0,3,0) - (0,1,1,3,3) + (0,0,1,0,3)$.

Now let C be an unauthorized set. Then we have the following possible cases for C:

Case 1:$|C| = 1$, then it is obvious that none of the rows of M is $(1,0,0,0,0)$.

Case 2:$|C| = 2$, then here is the list of unauthorized such sets:

$\{P_1, P_2\}, \{P_1, P_3\}, \{P_1, P_4\}, \{P_1, P_5\}, \{P_1, P_6\}, \{P_2, P_3\}, \{P_2, P_4\}, \{P_2, P_5\},$
$\{P_2, P_6\}, \{P_3, P_4\}, \{P_3, P_5\}, \{P_3, P_6\}.$

Let's check for $\{P_1, P_2\}$: $(1,0,0,0,0) = a_1(1,1,0,2,0) + a_2(0,1,1,2,2)$.

$a_1 = 1$, $a_2 = -1$ and 0, thus it is a contradiction.

Hence $\{P_1, P_2\}$ cannot generate $(1,0,0,0,0)$. The other cases are left to the reader to test.

Case 3:$|C| = 3$, then here is the list of unauthorized such sets:

$\{P_1, P_2, P_4\}, \{P_1, P_2, P_5\}, \{P_1, P_2, P_6\}, \{P_1, P_3, P_4\}, \{P_1, P_3, P_5\}, \{P_1, P_3, P_6\},$
$\{P_1, P_4, P_5\}, \{P_1, P_5, P_6\}, \{P_1, P_5, P_6\}, \{P_2, P_4, P_5\}, \{P_2, P_5, P_6\}, \{P_3, P_4, P_5\},$
$\{P_3, P_5, P_6\}$.

Let's check for $\{P_1, P_2, P_4\}$: $(1,0,0,0,0) = a_1(1,1,0,2,0) + a_2(0,1,1,2,2) + a_3(1,1,0,3,0)$. $a_2 = 0, a_1 + a_3 = 1$ and $a_1 + a_3 = 0$, thus it is a contradiction. Hence $\{P_1, P_2, P_4\}$ cannot generate $(1,0,0,0,0)$. The other cases are left to the reader to test.

Case 4:$|C| = 4$, then here is the list of unauthorized such sets:

$\{P_1, P_2, P_4, P_5\}, \{P_1, P_2, P_5, P_6\}, \{P_2, P_3, P_4, P_5\}, \{P_1, P_2, P_5, P_6\}.$

Let's check for $\{P_1, P_2, P_4, P_5\}$:

$$(1,0,0,0,0) = a_1(1,1,0,2,0) + a_2(0,1,1,2,2) + a_3(1,1,0,3,0) + a_4(0,1,1,3,3)$$

$$= a_1(1,1,0,2,0) + b(0,2,2,5,5) + a_3(1,1,0,3,0)$$

If and only if

$$a_1 + a_3 = 1,\ a_1 + b + a_3 = 0, 2b = 0$$

It is a contradiction. ■

Theorem 2.0: Let $\mathcal{P} = \{P_1, P_2, P_3, \dots, P_n\}$ be set of participants and $\Gamma = \{B_1, B_2, \dots, B_k\}$ be an access structure with the following two properties:

- $B_i \cap B_j = \emptyset\ for\ all\ i \neq j.$
- $|B_j| = m\ for\ j = 1, 2, 3, .. k.$

Then matrix M constructed by Algorithm 3.1 satisfies the (2.3) property.

Proof: First one can observe that the matrix M_i constructed in step 1-4 has the following properties:

1) First column of M_i has only one 1.
2) Columns 2 to m have exactly two 1's.
3) Columns *m+1* to *2m-1* have exactly two x_i's.

Let B_i be an authorized set. Then share of participants of B_i are rows of the matrix M_i. Let b_{i_t}, where $t = 1,2, \dots, m$ be rows of M_i. Then we have the following from the properties of M_i:

$$(1,0,0, \dots ,0) = \sum_{t=1}^{m} a_t \, b_{i_t} = (b_{i_1} + b_{i_3} + \cdots) - (b_{i_2} + b_{i_4} + \cdots)$$

That is, $(1, 0, 0, \dots., 0)$ can be created as a linear combinations of shares of authorized set B_i. Next let C be an unauthorized set. Then we need to show that the shares of C cannot generate $(1,0,0,..,0)$ of their linear combination. Now we have the following cases:

Case 1: $|C| = s < m$ Then the C is either a subset of an authorized set or C contains participants from at least two different authorized set. First assume C is a subset of an authorized set. Then there exist i such that $C \subset B_i$. Let the rows of matrices corresponding shares of C be b_{i_j}, where $1 \leq j \leq m$. Since $|C| < m$, there is at least one row missing, say b_{i_t}, where $1 \leq t \leq m$. If the first row is missing, then there is no way to form $(1,0,0, \dots ,0)$. Otherwise

$$\begin{aligned}
(1,0,0, \dots ,0) = \sum_{r=1}^{s} a_r \, b_{i_r} &\Leftrightarrow a_1 = 1 \\
&\Leftrightarrow a_1 + a_2 = 0 \\
&\quad \dots. \qquad \dots. \\
&\Leftrightarrow a_{t-2} + a_{t-1} = 0 \\
&\Leftrightarrow a_{t-1} = 0 \\
&\Leftrightarrow a_{t+1} = 0 \\
&\Leftrightarrow a_{t+1} + a_{t+2} = 0
\end{aligned}$$

$$\cdots \quad \cdots.$$

$$\Leftrightarrow a_{s-1} + a_s = 0$$

But this yields a contradiction because $a_{t-1} = 0$ and 1(or -1). If C contains participants from at least two different authorized set, then C may contain rows which are the same for the first m columns (see Case 4 in Example 3.2.4). We can add these rows to create a single row that has two non-zero elements in corresponding columns. Then the corresponding matrix will look like this:

y_1	y_1	0	…	0	$x`_1$	0	…	0
0	y_2	y_2	…	0	$x`_2$	$x`_2$	…	0
…	…	…	…	…	…	…	…	…
0	0	0	…	y_s	0	0	…	$x`_s$

Hence $(1,0,0,\ldots,0) = \sum_{r=1}^{s} a_r\, b_{i_r} \Leftrightarrow a_1 y_1 = 1$

$$\Leftrightarrow a_1 y_1 + a_2 y_2 = 0$$

$$\cdots \quad \cdots$$

$$\Leftrightarrow a_{s-1} y_{s-1} + a_s y_s = 0$$

$$\cdots \quad \cdots$$

$$\Leftrightarrow a_{s-1} x`_{s-1} + a_s x`_s = 0$$

By $a_1 y_1 = 1$, $a_1 = {y_1}^{-1}\ mod\ p$, hence all the a_i can be uniquely determined. But $x_i` \neq y_i\ mod\ p$ for all $i = 1,2,3,\ldots,s$ since $p$ is large enough and by $x_i \in Z_p$ such that $x_m > x_{m-1} > \cdots > x_1 > 1$. Therefore $a_i$ cannot satisfy $a_{s-1} x`_{s-1} + a_s x`_s = 0$.

Case 2: $|C| = s = m$ Then C shares participants from at least two different authorized sets B_i and B_j where $i \neq j$. Suppose $(1,0,0,\ldots,0)$ is a linear combination of secret shares of participants in C.

Then we have $(1,0,0,\ldots,0) = \sum_{r=1}^{s} a_r\, b_{k_r} \Leftrightarrow a_1 = 1, a_1 + a_2 = 0, \ldots, a_{s-1} + a_s = 0$ and there will be an index r for which $a_r(x_i - x_j) = 0$, where $a_i \in Z_p$. But this yields a contradiction because a_s=0 or 1(or -1) but x_i and x_j belonging to B_i and B_j are distinct and bigger than 1 as at least two participants from B_i and B_j. Hence C cannot generate $(1,0,0,\ldots,0)$.

Case 3: $|C| = s > m$. Here the number of rows cannot be greater than m. We refer case 1 and 2 to provide proof.

Hence C cannot generate $(1,0,0,\ldots,0)$.

3.4 Brute Force Search

In this section we tabulate some brute force test results. We wrote a brute force algorithm for small parameters to construct □ function.

System Configuration which was used to run the brute force search algorithm:

System Type	It ran on a node (of a server) with 16 cores
RAM	96 GB of RAM
Operating System	Red Hat 5.6 64-bit
CPU	Intel(R) Xeon(R) CPU E5640 @ 2.67GHz

Table- 3 System configuration used for Brute force search

Programming language and other tools used to develop and run the brute force search:

Programming language	Java 1.6.0_21
External Tools	Mathematica 8.0.0.0

Table- 4 Programming Language and tools used for Brute force search

The following are the different cases for which the brute force was performed.

Case 1: *Input*: $n := 4, m := 2, column := 2 * m - 2 = 2 \; and \; p = 3.$

Number of matrices tested: $3^8 = 6561.$

Time Taken: 10 seconds.

Result: no such function.

Case 2: *Input:* $n := 6, m := 2, column := 2 * m - 2 = 2 \; and \; p = 3.$

Number of matrices tested: $3^{12} = 531441.$

Time Taken: 3600 seconds.

Result: no such function.

Case 3: *Input*: $n := 6, m := 2, column := 2 * m - 1 = 3 \; and \; p = 3.$

Number of matrices tested: $3^{18} = 387420489.$

Time Taken: 70200 seconds.

Result: no such function.

Case 4: *Input*: $n \coloneqq 6, m := 3, column := 2 * m - 2 = 4 \; and \; p = 3.$

Number of matrices tested: $3^{24} = 282429536481.$

Time Taken: 518400 seconds.

Result: no such function.

CHAPTER 4: VISUAL CRYPTOGRAPHY

In this chapter, we discuss implementation of some secret sharing scheme on Visual Cryptography. Visual Cryptography was first introduced by Moni Naor and Adi Shamir in 1994 [9]. They demonstrated a visual secret sharing scheme, where an image was broken up into n shares so that only someone with all n shares could decrypt the image, while any $n - 1$ shares revealed no information about the original image.

Definition 4.0: Visual Cryptography is a method through which visual information like (image, video …) are encrypted and decrypted to receive original visual information. Let's illustrate, through two examples.

Example 4.0: A visual file (browser window) [16] which consists of pixels colored in black and white. The encryption is performed by hiding the image behind the pixels. To decrypt the image, one has to adjust the size of the file to a defined dimension; so that the image is visible by grouping of the pixels. This is one way how visual file is encrypted and decrypted. Here the secret key K is the dimension of the file.

The dimensions of the visual file are:

Width: 1366 px.

Height: 667 px.

The key dimensions through which the image can be visible are:

Width: 798 px.

Height: any px

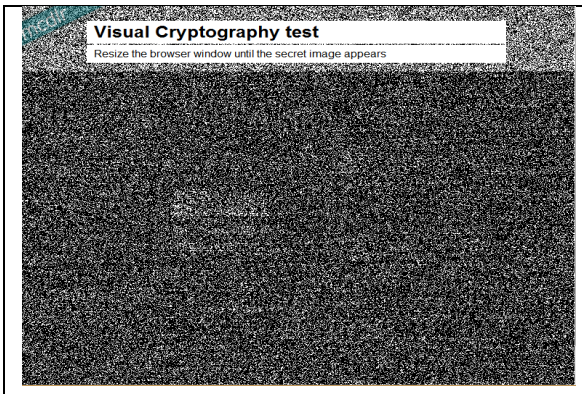	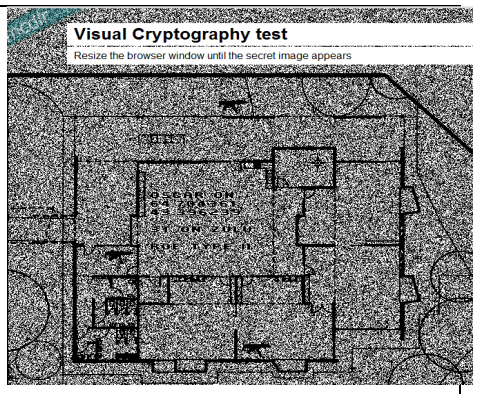
Image encrypted behind the pixels	Image is decrypted by providing the key

Figure- 10. Visual Cryptography■

The Visual Cryptography technique which we use in this thesis is different from what we have seen in previous examples. Following is an algorithm which we can use to encrypt and decrypt visual file (image).

***Algorithm 4.0*:** *Algorithm to shuffle and reshuffle image, based on permutation*

Input: $Image$, $permutation[n]$, $rows$ and $columns$

Output: Shuffled Image S.

Initial process:

1. Calculate $n := rows * columns$;
2. Calculate $chunkheight \ := \ imageheight \ / \ rows$;

$$chunkwidth \coloneqq imagewidth \,/\, columns\ ;$$

x_i - are the chunks of the Image $where\ 0 < i < n$.

3. // Breaking image into n small chunks

   $for\ i := 0\ to\ n$

   x_i := image with dimension chunkwidth, chunkheight.

4. $for\ i := 0\ to\ n$ // assign the image chunks to array chunkimage[ ]

   $chunkimage[i] = x_i$;

Shuffling of Image:

1. $for\ i := 0\ to\ n$

   $temp[i] \coloneqq chunkimage\big[permutation[i]\big]$;

2. $S \coloneqq draw\ (temp)$ // on Image

Re-Shuffling of Image:

1. Find inverse permutation for permutation.

2. $for\ i := 0\ to\ n$

   $temp[i] \coloneqq chunkimage\big[inversepermutation[i]\big]$;

3. $S := draw(temp)$;

Example 4.0.1: Consider an image X, rows=2, column=2 and permutation [] = {3, 1, 4, 2}

x_1	x_2
x_3	x_4

Figure- 11. Original Image X

Step 1: // Assume Image width=500, height=500

Step 2: // Calculate $n = rows * column$ that is $n = 4$.

Step 3: // Calculate $chunkwidth = \frac{imagewidth}{rows} = \frac{500}{2} = 250$ and $chunkheight = \frac{imageheight}{column} = \frac{500}{2} = 250$.

Step 4: Total no of chunks possible is $x_i: 1 \leq i \leq 4$.

Step 5: Now break the image into 4 pieces (chunks) and assign to corresponding x_i.

Step 5: $chunkimage[i] := x_i$ // array chunkimage is initialized with x_i.

Step 6: Based on permutation the image is shuffled by shifting the positions of the chunks in the array. So from given permutation {3, 1, 4, 2}:

1	2	3	4
3	1	4	2

$1 \rightarrow 3,\ 2 \rightarrow 1,\ 3 \rightarrow 4,\ 4 \rightarrow 2$. The image S holds the chunks with its new positions.

x_2	x_4
x_1	x_3

Figure- 12. Encrypted Image S.

Step 7: To decrypt: First one needs to find the inverse for given permutation. Based on given permutation $inversepermutation[\,] = \{2, 4, 1, 3\}$.

1	2	3	4
3	1	4	2

Step 8: The $inversepermutation[\,]$ is applied on encrypted image S, such that

$1 \rightarrow 2,\ 2 \rightarrow 4,\ 3 \rightarrow 1,\ 4 \rightarrow 3$, thus giving back the original image.

x_1	x_2
x_3	x_4

Figure- 13. Decrypted Image S ■

In visual cryptography application the secret sharing scheme is not the main concern as any of the schemes can be used for key share distribution. This visual application uses key as a permutation, which is not integer but somehow we need to relate integer to

permutation. One possible way which I can think of is converting an integer to its corresponding permutation and vice versa.

Example 4.0.2: Encryption and decryption of image based on the above example permutation.

Permutation = {3, 1, 4, 2}

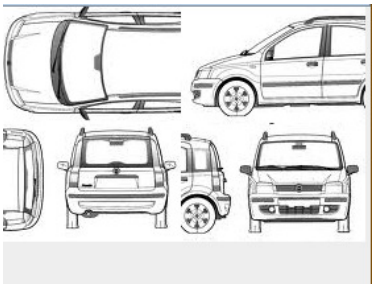

Figure- 14. Image encrypted

Inverse permutation= {2, 4, 1, 3}

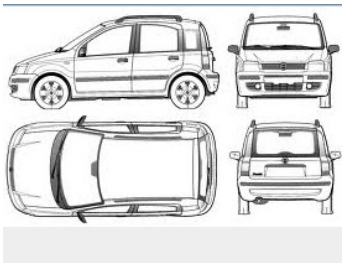

Figure- 15.Image decrypted ■

4.1Generating a permutation from secret key *K*

In all previous secret sharing schemes, secret key K is an integer but this visual crypto system uses permutation as a secret key value. We need to able to find one-to-one relation between given integer and permutation, so that we can use any one of these

secret sharing scheme in our visual crypto system. Before we give an algorithm to construct a permutation for any given key K, we illustrate this construction with an example.

Example 4.1.1: Illustration of permutation generation from secret key K.

Step 1: Select $p \coloneqq 137$ a prime number, so $Z_{137} = \{0, \dots, 136\}$.

Step 2: Now find a smallest n value such that $p \leq n!$. Hence $n \coloneqq 6$.

Step 3: Now K is selected, such that $K \in Z_p$.

Step 4: Let suppose, $K \coloneqq 53$.

Step 5: Generate all possible permutation of $\{1,2,3,4,5,6\}$ in lexical order.

Step 6: From the list of permutations, the corresponding permutation for $k \coloneqq 53$ will be 53rd permutation of 6!

Step 7: Else one can use an algorithm which relates an integer to permutation ; which is discussed in **Algorithm 4.1.1**.■

Algorithm 4.1.1[17]**:** *To generate permutation from integer*

Input: K where $K \in Z_p$ and n $where$ $p \leq n!$.

Initial process: Consider a string variable $a \coloneqq [1234 \dots n]$.

The kth permutation starts from$\{0,1, \dots, n-1\}$.

Working:

Step 1: Find the $a_1, a_2, \ldots, a^{th}{}_n$ character of string a from

$k - 1 = a_1(n-1)! + a_2(n-2)! + \cdots + a_n.0!$ such that

$0 \leq a_i \leq n - i.$

Step 2: Then take the ${a_1}^{th}$ character in the string, followed by the ${a_2}^{th}$ character of what's left, etc.

Step 3: Thus bringing all the characters together will represent the permutation.

Example 4.1.2:

Input: $Let\ K = 21 \in Z_{119}$, so $n = 5$ as$119 < 5!$. To find 21st permutation, one needs to find 20th permutation, according to above algorithm. Let String $a = 12345$.

Step1: $20 = a_1(4)! + a_2(3)! + a_3(2)! + a_4(1)! + a_5.0!$. Now find $a_1, .. a_5$ values satisfying condition $0 \leq a_i \leq n - i$.

Step 2: The values are $a_1 = 0, a_2 = 3, a_3 = 1, a_4 = 0, a_5 = 0$

Step 3: Hence,

$a_1 = 0^{th}$ element of 12345 is 1 and remaining string is 2345

$a_2 = 3^{rd}$ element of 2345 is 5 and remaining string is 234

$a_3 = 1^{st}$ element of 234 is 3 and remaining string is 24

$a_4 = 0^{th}$ element of 24 is 2 and remaining string is 4

$a_5 = 0^{th}$ element of 4 is 4.

Step 4: So 20th permutation is 15324.

Step 5: Hence $K = 21 \rightarrow permutation\ 15324$ ■

CHAPTER 5: CONCLUSION

Brickell vector space secret sharing construction algorithm requires existence of function with certain conditions. There is no known algorithm to construct such function for any given access structure. Construction algorithm exists only if access structure is represented as an edge set of complete multipartite graph. For the rest, only trial and error is an option. We have developed a polynomial algorithm to construct such function for another special access structure which is a multipartite graph but not complete. Moreover we used a brute force test, to find any better construction with less columns on some of the small parameters. For example $n = 4, m = 2, p = 3$ and $n = 6$ with $m = 2,3$, $p = 3$. But there is no function in which matrix representation has a less number of columns than *2m-1*.

REFERENCES

[1] H.Anton "*Elementary Linear Algebra*", Eighth Edition. John Wiley and Sons, 2000.

[2] H.Beker and F.Piper. "*Cipher Systems, The Protection of Communications"*. John Wiley and Sons, 1982.

[3] G.R. Blakley. Safeguarding cryptographic keys. *Federal Information Processing Standard Conference Proceedings*, **48** (1979), 313-317.

[4] E. F. Brickell. Some ideal secret sharing schemes. *Journal of Combinatorial Mathematics and Combinatorial Computing*, **9** (1989), 105-113.

[5] E. F. Brickell and D. M. Davenport. On the classification of ideal secret sharing schemes. *Journal of Cryptology*, **4** (1991), 123-134.

[6] Marten van Dijk. *Designs, codes and Cryptography*. 1997, Volume 12, Number 2, pg.161-201.http://www.mendeley.com/research/linear-construction-secret-sharing-schemes/.

[7] M. ITO, A. Saito and T.Nishizeki. Secret sharing scheme realizing general access structure. Proceedings *IEEE Globecom* '87, pages 99-102, 1987.

[8] A.G. Konheim. "*Cryptography"*, A Primer. John Wiley and Sons, 1989.

[9] Moni Naor and Adi Shamir. *Visual Cryptography*. EuroCrypt'94. http://www.wisdom.weizmann.ac.il/~naor/PUZZLES/visual_sol.html.

[10] A Shamir. How to share a secret. *Communications of the* ACM, 22(1979), 612-613.

[11] G. J. Simmons. An introduction to shared secret and/ or shared control schemes and their application. In *Contemporary Cryptology, The Science of Information Integrity*, pages 441-497. IEEE Press, 1992.

[12] G. Shanker Rao. *Discrete Mathematics Structures*. NEW AGE INTERNATIONAL (P) LIMITED PUBLISHER, 2000.

[13] D.R Stinson "*Cryptography: theory and practice*" CRC, 1995.

[14] D.R Stinson "*Cryptography: theory and practice"* CRC, 2000.

[15] Time Magazine, May 4, 1992, p.13.

[16] http://mcdlr.com/visual-crypto/.

[17] http://www.ocf.berkeley.edu.