Semester Project

Secret Sharing Schemes

Avani Maroo, Veda D

Computer Security and Privacy (CS-2362)
Ashoka University
Date: May 1, 2022

Contents

1 Project Description

Through this project, we aim to understand the theoretical and mathematical aspects and do a simple implementation of two secret sharing schemes proposed by Adi Shamir : **Shamir Secret Sharing** and **Visual Secret sharing**.

What is Secret Sharing?

Secret Sharing Schemes are cryptographic methods which take a secret key (or what is normally considered a private key) and break it down into multiple shares. Each share is given to a trusted source. The key can be reconstructed only if a subset of the trusted sources come together with their respective shares of the key.

This can more rigorously defined as the following:

A "t-out-of-n threshold secret sharing scheme" (TSSS:) consists of the following algorithms:

1. **Share**: A randomized algorithm that takes a message $m \in M$ as input and outputs a sequence $S = (s_1, ..., s_n)$ of shares.

2. **Reconstruct**: A deterministic algorithm that takes a collection of t or more shares as an input and outputs a message. We can call M the message space of the scheme and t it's threshold.

The n parties that have received these shares s_i where $i \in \{1, ..., n\}$. Let there be a subset of users, $A \subseteq \{1, ..., n\}$. Essentially $\{s_i \mid i \in A\}$ refers to the shares which belong to the user A. A is **authorized** if $|A| \geq t$, else it is **unauthorized**.

What is the motivation behind secret sharing?

It was implemented with the intention of allowing more secure storage of very sensitive data such as encryption keys, missile launch keys and bank account numbers. It is currently found in cloud computing environments because of the level of security it provides. This is because by distributing shares of the message, there will no single point of failure due to the existence of a threshold.

2 Project Goals

1. **Shamir Secret Sharing**
   - Theoretical and Mathematical Understanding of Shamir Secret Sharing
   - Implementation of Shamir Secret Sharing
   - Implementation of short shares on Shamir Secret Sharing
2. **Visual Secret Sharing**
   - Theoretical and Mathematical Understanding of Visual Secret Sharing
   - Implementation of Visual Secret Sharing Scheme

3 Shamir Secret Sharing

A big foundation of Shamir Secret Sharing is based on the foundation of Polynomial Interpolation and Lagrange Multipliers. Therefore, before this report discusses the theoretical understanding of how Shamir Secret Sharing works, it would be important to understand the mathematical concepts behind it.

3.1 Polynomial Interpolation and Lagrange Multipliers

Theorem(Polynomial Interpolation): Let $\{(x_1, y_1), (x_2, y_2), ..., (x_{d+1}, y_{d+1})\} \subseteq \mathbb{R}^2$ be a set of points whose x_i values are all distinct. Then there can be a unique degree-d polynomial h that can be constructed that satisfies $y_i = h(x_i)\ \forall i$

Proof: Consider the polynomial

$$p_1(\bar{x}) = \frac{(\bar{x} - x_2)(\bar{x} - x_3)...(\bar{x} - x_{d+1})}{(x_1 - x_2)(x_1 - x_3)...(x_1 - x_{d+1})}$$

The numerator of the polynomial will be a d-degree polynomial while the denominator will be a scalar. Since all the x_i's are unique, the denominator will never be a 0. Therefore, $p_1(\bar{x})$ is a d-degree polynomial.

Let us consider what happens when we consider $\bar{x}$ to be any of the x_i's:

- **Case 1**: $p_1(\bar{x} = x_i) = 1$ where $i = 1$.This is because when $\bar{x}$ is substituted by x_1, the numerator is the same as the denominator.
- **Case 2**: $p_1(\bar{x} = x_i) = 0$ where $i \in \{2, 3, ..., d + 1\}$. This is because one of the $(\bar{x} - x_i)$ would be equal to 0, leading the numerator to be 0.

The above polynomial can also be evaluated at other points but they are not in interest or contribute to the knowledge of finding the unique $d-$degree polynomial that can be constructed from the $d + 1$ points.

Other $p_j(\bar{x})$ polynomials can be defined in the following manner:

$$p_j(\bar{x}) = \frac{(\bar{x} - x_1)(\bar{x} - x_2)...(\bar{x} - x_{d+1})}{(x_j - x_2)(x_j - x_3)...(x_j - x_{d+1})}$$

These polynomials are called **Lagrange Polynomials**. Each Lagrange polynomial is $d-$degree polynomials that satisfy the following property:

$$p_j(x_i) = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{if } i \neq j \end{cases}$$

Consider the following polynomial:

$$h(\bar{x}) = y_1 p_1(\bar{x}) + y_2 p_2(\bar{x}) + ... + y_{d+1} p_{d+1}(\bar{x})$$

The polynomial h is a sum of $d-$degree polynomials.When $h(x_i)$ is evaluated where $i \in \{1, 2, .., d+1\}$, it would be equal to y_i. It could formally be seen as:

$$h(x_i) = y_1p_1(x_i) + y_2p_2(x_i) + ... + y_ip_i(x_i) + ... + y_{d+1}p_{d+1}(x_i)$$

$$h(x_i) = y_1 \cdot 0 + y_2 \cdot 0 + ... + y_i \cdot 1 + ... + y_{d+1} \cdot 0$$

$$h(x_i) = y_i$$

The substitution can be derived from the property of Lagrange Polynomials. Therefore, $h(x_i) = y_i$ as we would like the points x_i's to lie on the polynomial h.

Now to prove that the above polynomial h is unique $d-$degree polynomial. This can be done via proof of contradiction. Suppose there are two d-degree polynomials h and h' such that $h(x_i) = h'(x_i) = y$ for $i \in \{1, 2, ..., d+1\}$. Let a polynomial g be constructed, such that: $g(\bar{x}) = h(\bar{x}) - h'(\bar{x})$, and g is a $d-$degree polynomial and it will satisfy $g(x_i) = 0 \forall i$. This means each x_i is a root of g, which means g has $d+1$ roots. However, the only $d-$degree polynomial with $d+1$ roots is the zero polynomial, $g(\bar{x}) = 0$.This implies that $h = h'$. Therefore, h must be a unique $d-$degree polynomial.

3.2 Introduction

To ensure there is no information leakage until the threshold number of shares of the secret are reached, the coefficients of the polynomial must be from $\mathbb{Z}_p$. Therefore, to share a secret message, $m \in \mathbb{Z}_p$, with threshold t. The polynomial chosen must be of degree $(t-1)$ polynomial h that satisfies $h(0) \equiv_p m$, with all other co-efficients chosen uniformly in $\mathbb{Z}_p$
i^{th} user receiver receives the following share as the point $(i, h(i)\%p)$ on the polynomial. Through the polynomial interpolation theorem present above, using any of the t shares, the polynomial can be reconstructed, and hence $h(0)$ which is equal to the secret message m which can be retrieved.

3.3 Construction of the Shamir Secret Sharing Algorithm

Sharing Algorithm

Let the algorithm have the input $m \in \mathbb{Z}_p$. Let $t-1$ numbers be randomly picked from Z_p where p is a prime number. The variable $t \leq n$ and $n < p$. Let the polynomial $h(\bar{x})$ be formally defined as:

$$h(\bar{x}) = m + \Sigma_{j=1}^{t-1} f_j \bar{x}^j$$

The set of points given to the user, is constructed in the following manner: $s_i = (i, h(i)\%p)$ where i is the identity of the user. The program returns all $s = (s_1, s_2, .., s_n)$.

Reconstruction Algorithm

Reconstruction($\{s_i | i \in A\}$) performs the following: $h(\bar{x})$, a unique $t-1-$degree polynomial mod p is constructed that ensures the polynomial passes through the points ($\{s_i | i \in A\}$). This can be done using Lagrange polynomials. The function returns $f(0)$.

Proof

The correctness of the theorem follows by Lagrange interpolation theorem and the uniqueness of the polynomial created.

3.4 Implementation

By studying the paper Secret Sharing Made Short by Hugo Krawcyzk, this project has an implementation of the method in Section 3: Secret Sharing with Short Shares. The scheme proposes combining Information Dispersal Algorithm with Shamir Secret Sharing. Using the Information Dispersal Algorithm proposed by Rabin, the scheme proposes to distribute the message amongst n users and in the presence of m users the message can be recovered.m and n follow the following parameters: $1 \leq m \leq n$. The scheme assumes that the users don't modify the fragments. Each fragment is of size $\frac{|F|}{m}$ for efficiency and reconstruction from m fragments is possible. Combining this with Shamir Secret Sharing, one can create a secure system of offering multiple shares of encrypted files to many users and recover them in the presence of the decided threshold shares.

The algorithm works in the following way:

Distribution Scheme

1. Choose a random key K. It is used to encrypt the message/file S, under the key. $E = ENC(S, K)$ where E is the encrypted message.

2. Using the Information Dispersal Algorithm, the encrypted file E, is partitioned into n shares that can be represented as $E_1, E_2, ..., E_n$

3. Using Perfect Secret Sharing(in the case of this project Shamir Secret Sharing), the Key K is partitioned into n shares that can be represented as $K_1, K_2, ..., K_n$

4. Each participant is given a share which is a two- tuple containing a fragment of the message and they key, $S_i = (E_i, K_i)$

Reconstruction Scheme

1. From m Participant P_{i_j} where $j = 1, 2, ..., m$ and their shares $S_{i_j} = (E_{i_j}, K_{i_j})$

2. Using the Information Dispersal Algorithm, the encrypted message/file E can be reconstructed from the collected values E_{i_j} where $j = 1, 2, ..., m$

3. Using Shamir Secret Sharing, the key K is recovered from K_{i_j} where $j = 1, 2, ...m$

4. The message/file E can be decrpyted by key K, to return the message file S, i.e $S = DEC(E, K)$

Limitation of implementation

A more specifically created Information Dispersal Algorithm, would allow rather a better way of taking the fragments of the file as currently they're encoded in a class created by the package with not a lot of details available regarding the package's meaning. The algorithmic implementation of this scheme also is currently not delivering the key fragment of the share in another private secure system.

4 Visual Secret Sharing

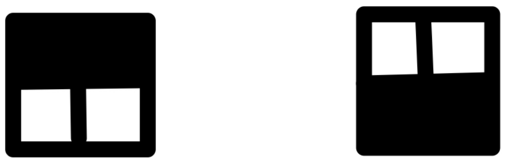
4.1 What is Visual Secret Sharing? (The Idea)

Visual secret sharing is a subtype of secret sharing schemes which was proposed by Adi Shamir and Moni Naor. It refers to an encryption technique where a secret image is broken into n shares and any $n-1$ shares are not enough for decryption. It could also be in the form of a threshold scheme, where a subset of the shares need to be superimposed in order to reconstruct the original image. The scheme that will be describing is 2-by-2 visual secret sharing. Here, the source image is divided into 2 shares. Each share should be printed on transparent sheets and when the two shares are stacked on top of each other, the secret image is revealed visually.

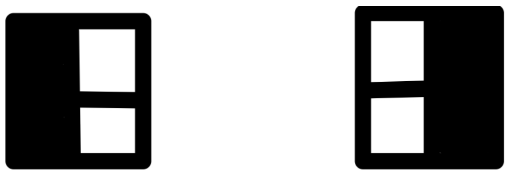
4.2 A Detailed Explanation of the Scheme

The assumption here that the secret image is black and white. Thus, there are two colours of pixels that need to be encrypted: black and white. For encryption, different ”pixel patterns” are used. These are as follows:

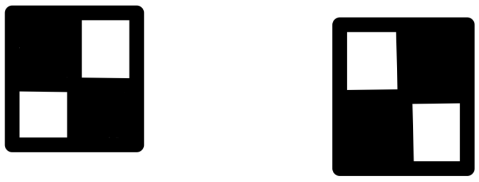
Horizontal Pattern:



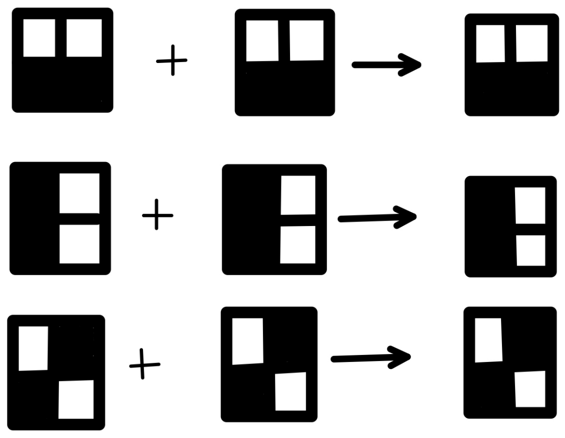
Vertical Pattern:



Diagonal Pattern:

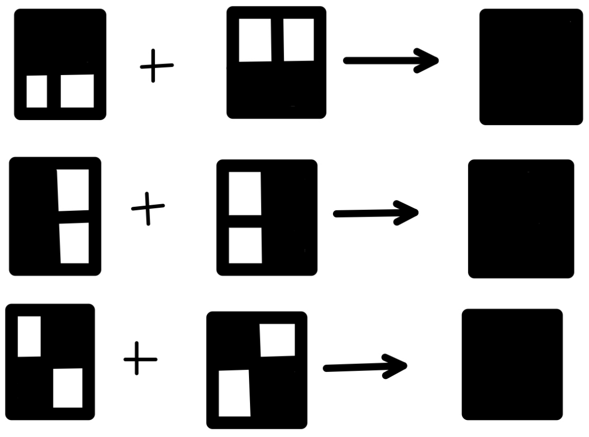


When two similar pixel patterns are stacked on one another, they give a similar output:



We call this output *gray*.

When two different pixel patterns are stacked on one another, they give different output:



We call this output *black*.

Essentially, we take each pixel of our secret message and encode each pixel as a 2x2 block of pixels in each of the shares. Since we have 2 shares here, we will encode each pixel into a suitable pixel pattern, twice. Since each pixel is replaced by a 2 x 2 pixel pattern block, the shares of the image and the decrypted image will be twice the size of the source image.

A white pixel is shared in the two shares, in a way that when the two images are stacked on top of one another, the output pixels in that place should be *gray*.
Similarly, a black pixel is shared in the two shares, in a way that when the two images are stacked on top of one another, the output pixels in that place should be *black*.

Essentially when the source pixel is white, the two shares have identical 2×2 blocks, so that when stacked, they make a *gray* block. Whenever a source pixel is black, the two shares have opposite blocks, so stack to make a black block.

4.3 Security of the Algorithm

The above described $(2, 2)$ scheme, is indeed secure. That is, looking at one share won't reveal any information about the secret image, not even at the pixel level. Since each pixel is split into 4 sub-pixels, with 2 white sub-pixels and 2 black sub-pixels, it is difficult to tell just by looking at one share's one sub-pixel if the corresponding pixel of the original image was black or white. Furthermore, the pixel pattern possibilities are equally likely to occur irrespective of whether the corresponding pixel in the original image is white or black. Thus, no information is obtained by looking at any group of pixels in a single share.

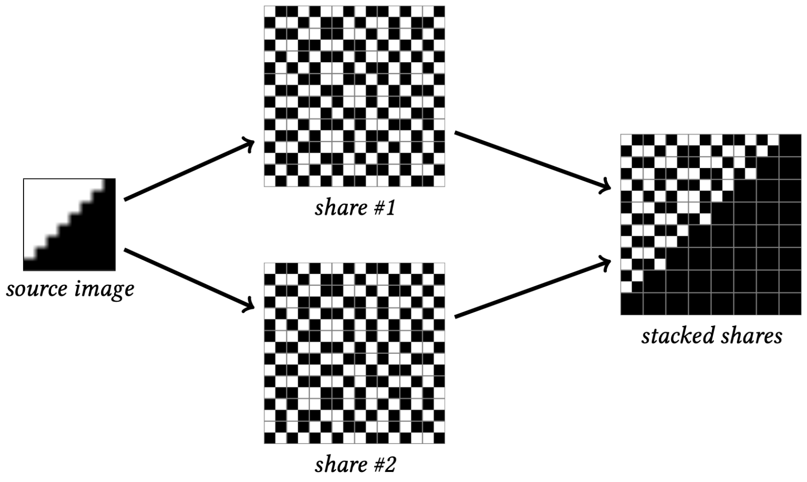
4.4 Implementation

A Simple Algorithm:

Share(m): initialize empty images s_1, s_2, with dimensions twice that of m for each position $(i, j) \in m$:

- randomly choose b1 to be any pixel pattern
- if $m[i, j]$ is a white pixel: set b_2 to be the same as $b1$
- if $m[i, j]$ is a black pixel: set b_2 to be the opposite of $b1$
- add 2 x 2 block b_1 to image s_1 at position $(2i, 2j)$
- add 2 x 2 block b_2 to image s_2 at position $(2i, 2j)$
- return (s_1, s_2)

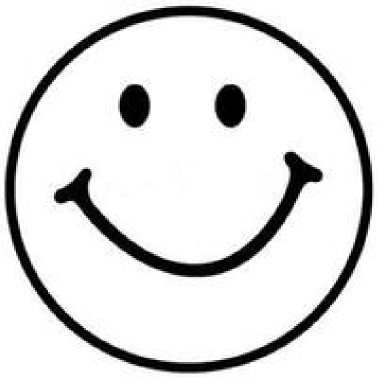
Essentially, the implementation process will do something like this:



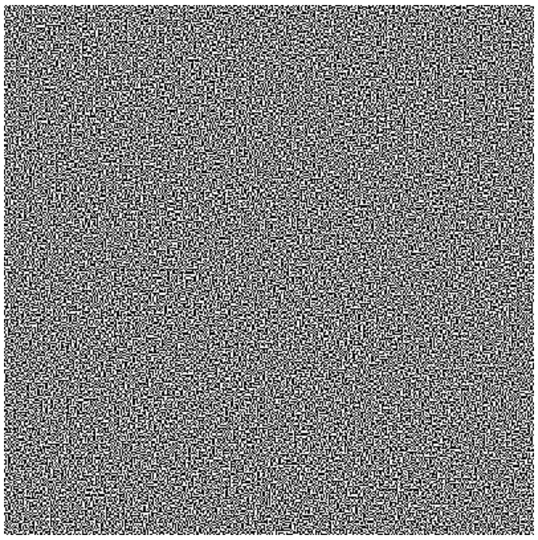
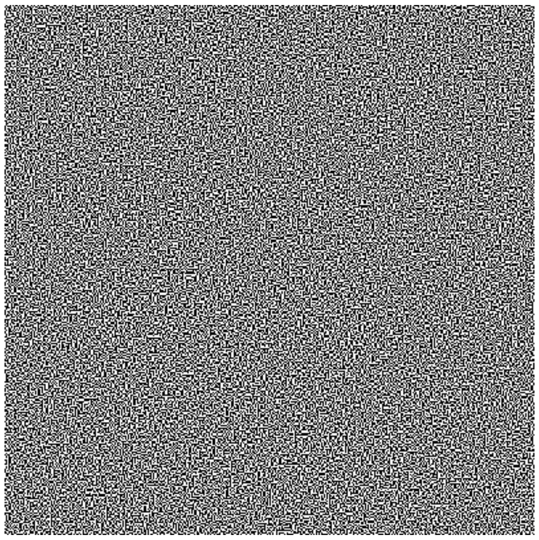
Code Implementation:

With the code implemented, here is how the given image will be spilt into shares:

Original Image:



Encrypted shares of the image:

Once these are printed on transparencies and stacked on one another, the decrypted image will be revealed.

5 Citations

1. Joy of Cryptography
2. Secret Sharing Made Short by Hugo Krawcy

6 GitHub Repo Link

`https://github.com/veda-duddu/CS2362-Project`