**Lecture Notes on SECRET SHARING**
**Exposition by Bill Gasarch**

# 1 Introduction

The field of Secret Sharing was invented by Adi Shamir [6] and George Blakely [2] independently.

Zelda has a secret $s$ which is a string of bits. She has associates Alice and Bob. She wants to give SOME info to Alice and SOME info to Bob (called *a share of the secret*) such that

- Alice alone has NO IDEA what the secret is (info-theoretic security).
- Bob alone has NO IDEA what the secret is (info-theoretic security).
- If Alice and Bob share their information then they can both learn the secret.

This problem can be generalized to Zelda having three friends and any TWO cannot uncover the secret, but all three CAN.

This problem can be generalized further: Zelda has many friends and she only wants certain subsets of them (and supersets of those) to learn the secret, but no proper subsets of those can learn the secret.

We will show how all of these things can be done.

# 2 Alice and Bob need to Cooperate to Learn the Secret

**Def 2.1** If $b$ and $c$ are bits (elements of $\{0,1\}$ then $\oplus$ (pronounced "x-or") is defined as follows

| $b$ | $c$ | $b \oplus c$ |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |

Note that $b \oplus c \oplus c = b$.

1. Zelda's secret is a string of bits $s_1 s_2 \cdots s_L$.
2. Zelda generates a RANDOM SEQUENCE OF $L$ bits: $a_1 a_2 \cdots a_L$.
3. Zelda computes

$$\begin{aligned} b_1 &= s_1 \oplus a_1 \\ b_2 &= s_2 \oplus a_2 \\ \vdots &\ \vdots \\ b_L &= s_L \oplus a_L \end{aligned}$$

4. Zelda gives Alice $a_1 a_2 \cdots a_L$.
5. Zelda gives Bob $b_1 b_2 \cdots b_L$.

Alice alone has $a_1 \cdots a_L$ which is a RANDOM sequence of bits. NO information.
Bob alone has $b_1 \cdots b_L$ which is a RANDOM sequence of bits. NO information.
But if they get together then they can XOR the strings bitwise to obtain

$$\begin{aligned} s_1 &= a_1 \oplus b_1 \\ s_2 &= a_2 \oplus b_2 \\ \vdots &\ \vdots \\ s_L &= a_L \oplus b_L \end{aligned}$$

# 3 Certain Subsets of Alice, Bob, Carol, Donna Need to Cooperate to Learn the Secret

**Notation 3.1** IMPORTANT NOTATION: If $x, y$ are strings of bits of the same length then $x \oplus y$ is the bit-wise $\oplus$ of $x$ and $y$.

Zelda has a secret. She wants it to be the case that:

- Alice, Bob, Carol together can crack it, but no subset.
- Carol, Donna together can crack it, but no subset.

We essentially do the protocol in the last section twice.

1. Zelda's secret is a string of bits $s$.
2. Zelda generates a RANDOM SEQUENCE OF BITS $a$, with $|a| = |s|$.
3. Zelda generates a RANDOM SEQUENCE OF BITS $b$, with $|b| = |s|$.
4. Zelda computes $c = s \oplus a \oplus b$.
5. Zelda gives Alice ($a$, ABC).
6. Zelda gives Bob ($b$, ABC).
7. Zelda gives Carol ($c$, ABC).

The ABC is telling them to use those strings JUST for Alice-Bob-Carol. If Alice, Bob, Carol get together note that

$$a \oplus b \oplus c = a \oplus b \oplus (a \oplus b \oplus s) = s$$

OKAY, that takes care of Alice, Bob, Carol. We now want to take care of Carol and Donna. We will call the bits we give Carol $c'$ to distinguish them for the $c$.

1. Zelda's secret is a string of bits $s$.
2. Zelda generates a RANDOM SEQUENCE OF of bits: $c'$ with $|c'| = |c|$.
3. Zelda computes $d = s \oplus c'$.
4. Zelda gives Carol ($c'$,CD).
5. Zelda gives Donna ($d$,CD).

The CD is telling them to use those strings JUST for Carol-Donna. If Carol and Donna get together note that

$$c' \oplus d = c' \oplus (s \oplus c') = s.$$

# 4 Specified Subsets of $A_1, \ldots, A_m$ have to Cooperate to Learn the Secret

Assume the secret is a string $s \in \{0,1\}^L$.

What if the people are $A_1, \ldots, A_m$ and the subsets of people that you want to allow to have $S_1, \ldots, S_L$. Assume the secret is of length $L$.

1. Assume $S_i = \{A_1, \ldots, A_z\}$ after renumbering.
2. Zelda generates random strings $r_1, \ldots, r_{z-1} \in \{0,1\}^L$.

3. Zelda gives each of $A_1, \ldots, A_{z-1}$ a random string of $L$ bits.

4. Zelda gives $A_z$ that $\oplus$ of ALL of the strings given to $A_1, \ldots, A_{z-1}$ together with the number $i$ (so that they know the string they got is to be used when the people in $S_i$ are to find the secret). and then $\oplus$ that with the secret.

.

GOOD NEWS: This always works!

BAD NEWS: Zelda is giving out LOTS of strings. We do an example.

**Example:** There are 8 people and any subset of 4 should be able to crack the secret. The secret is of length $L$.

The number of subsets of $\frac{8!}{4!4!} = \frac{8\times7\times6\times5}{4\times3\times2} = 7 \times 2 \times 5 = 70$.

So there are 70 subsets. Each has 4 people. So Zelda is giving out $70 \times 4 = 280$ strings of length $L$.

We want to do better!

# 5 Threshold Secret Sharing

Section 4 had good news and bad news. The good news is that for ANY specified subsets Zelda can share her secret. The bad news is that it might involve many strings. We want to use far less strings. We formalize this

**Def 5.1** A secret shareing scheme is *ideal* if each $A_i$ gets a share of length the same as the length of the secret.

There are some subsets of $\{A_1, \ldots, A_m\}$ such that ideal secret sharing is not possible. Hence we look at a particular type of subsets where it is possible. Let $2 \leq t \leq m$. We want a secret sharing scheme such that the following happens:

- ANY $t$ of $A_1, \ldots, A_m$ can find out the secret.
- NO subset of size $t-1$ of $A_1, \ldots, A_m$ can find out the secret.

## 5.1 Digression into Abstract Algebra

Recall the domain $\{0, 1, \ldots, p-1\}$ where the mathematics is mod p. Here are some properties that this domain has:

1. There is an element 0 such that for all $x$, $x + 0 = x$.
2. There is an element 1 such that for all $x$, $x \times 1 = x$.
3. For all $x, y$, (1) $x + y = y + x$, and (2) $xy = yx$.
4. For all $x, y, z$ (1) $x + (y + z) = (x + y) + x$, and (2) $(xy)z = x(yz)$.
5. For all $x$ there exists $y$ such that $x + y = 0$. (So, for example, $-7$ makes sense.)
6. IF $p$ is prime then for all $x$ there exists $y$ such that $xy = 1$ (So, for example, $\frac{1}{7}$ makes sense.)

We use these properties to define a field.

**Def 5.2** A *Field* is a set of elements $F$ together with two operations + and $\times$ that satisfy the properties above.

Examples:

1. $\mathsf{Q}$, $\mathsf{R}$, $\mathsf{C}$ are fields you have seen in high school. Note that $\mathsf{N}$ and $\mathsf{Z}$ are NOT fields since you cannot divide.

2. Let $GF(p)$ be the set $\{0, 1, \ldots, p-1\}$ using mod $p$ arithmetic is a field. If $p$ is a prime then $GF(p)$ is a field. The only hard step to proving that is that every number has a mult inverse, which you have already proven. If $p$ is NOT a prime then $GF(p)$ is NOT a field. (Nobody ever uses the notation $GF(p)$ in this case since GF stands for Galois Field.)

Are there any other finite fields? YES. We need the following two theorems from abstract algebra.

**Theorem 5.3** *If $q$ is a power of a prime then there exists a unique field on $q$ elements. If $q$ is a NOT power of a prime then there DOES NOT exists a field on $q$ elements.*

**Notation 5.4** If $q$ is a finite field then $GF(q)$ is the finite field on $q$ elements.

**Theorem 5.5** *If $F$ is any field and $f(x)$ is a poly of degree $t-1$ over that field then (1) given $t$ values of $f$ (e.g., $f(1), \ldots, f(t)$) you can determine the polynomial, (2) given $t-2$ values of $f$ you cannot determine ANYTHING about the polynomial.*

**Proof:** We give two proofs of (1). We do not proof (2).

**Proof 1**

Let $1 \leq j \leq t$. Consider the function

$$h_j(x) = \frac{x-x_1}{x_j-x_1}\frac{x-x_2}{x_j-x_2}\cdots\frac{x-x_{j-1}}{x_j-x_{j-1}}\frac{x-x_{j+1}}{x_j-x_{j+1}}\frac{x-x_{j+2}}{x_j-x_{j+2}}\cdots\frac{x-x_t}{x_j-x_t}.$$

Note that

- For all $x \in \{x_1, \ldots, x_t\} - \{x_j\}$, $h_j(x) = 0$.
- $h_j(x_j) = 1$

.

We can use these polynomials to form the polynomial

$$f(x) = \sum_{j=1}^{t} y_j h_j(x).$$

Clearly, for all $1 \leq i \leq t$, $f(x_j) = y_j$. Also clearly $f$ is of degree $t-1$.

**Proof 2**

This method only works if the the the points are $(0, y_0), \ldots, (t-1, y_{t-1})$. This will NOT be useful for us since, as we will see later, we can't possibly give anyone $f(0)$ as that IS the secret. Even so, this method of interpolation will be useful for a later protocol.

Assume the polynomial is of the form

$$f(x) = c_0\binom{x}{0} + c_1\binom{x}{1} + \cdots + c_{t-1}\binom{x}{t-1}.$$

Note that if $y \geq x+1$ then $\binom{x}{y} = 0$.

First look at $f(0) = y_0$. This means

$$y_0 = c_0\binom{0}{0} = c_0.$$

So we know $c_0$.

Now look at $f(1) = y_1$. This means

$$y_1 = c_0\binom{1}{0} + c_1\binom{1}{1} = c_0 + c_1.$$

Hence $c_1 = y_1 - c_0$.
Now look at $f(2) = y_2$. This means

$$y_2 = c_0\binom{2}{0} + c_1\binom{2}{1} + c_2\binom{2}{2} = c_0 + c_1 + c_2.$$

Hence $c_2 = y_2 - (c_0 + c_1)$.
More generally we have:

$$(\forall 0 \leq i \leq t-1)[c_i = y_i - \sum_{j=0}^{i-1} c_i.$$

∎

We will prove this later.

### 5.2 Back to Threshold Secret Sharing

1. Zelda's secret is a string of bits $s$. Let $F = GF(2^{|s|})$. All arithmetic will be in $F$. Note that $s$ is an element of $F$. (NOTE- could also us a prime $p$ slightly bigger than $s$ and use $GF(p)$. This will lead to the shares of the secret being one bit longer, but the arithmetic, mod $p$, is more familiar to you.)

2. Zelda picks random $r_1, \ldots, r_{t-1} \in F$. Let $f$ be the function

$$f(x) = r_{t-1}x^{t-1} + \cdots + r_1 x + s.$$

3. For $1 \leq i \leq m$ Zelda gives $A_i$ the string $f(i)$. There are elements of $F$ and hence $|s|$ long. Note that these are $m$ points on the curve $f(x)$,

If any $t$ people get together then they can deterimine $f(x)$ and hence $s$. If any $t-1$ people get together they cannot determine ANYTHING about $s$. Note that all of the shares are of size $|s|$.

The key to the technique in this chapter is that $t$ points determine a degree $t-1$ polynomial. This is Shamir's [6] scheme. Blakely [2] used that $t$ points determine a $t-1$-space (e.g., three points determine a plane). We will present Shamir's protocol but not Blakely's. There is no good reason for this— its just that I like using polynomials rather than high-dimensional spaces.

## 6 Some Secret Sharing Schemes for Non-Threshold Structures

In Section 4 we saw how Zelda could share here secret with ANY set of subsets, though the schemes were far from ideal. In Section 5.2 we saw that if the goal is threshold then Zelda and company can do ideal secret sharing. Are there other sets of subsets where they can Zelda and friends can do ideal secret sharing.

### 6.1 Need $A_1, A_2, \ldots, A_{t'}$ and at least $t$ others

Like the title says.

1. $s$ is the secret.

2. Zelda picks random $s_1, \ldots, s_{t'}$ of the same length as $s$. and then finds $s'$ such that $s_1 \oplus \cdots \oplus s_{t'} \oplus s' = s$.

3. For $1 \leq i \leq t'$ Zelda gives $A_i$ the string $s_i$.

4. Zelda picks random $r_1, \ldots, r_{t-1} \in F$. Let $f$ be

$$f(x) = r_{t-1}x^{t-1} + \cdots + r_1 x + s'.$$

For $t'+1 \leq i \leq m$ Zelda gives $A_i$ the number $f(i)$. (This is just the usual poly-threshold scheme for $A_{t'+1}, \ldots, A_m$ with threshold $t$.

If $A_1, \ldots, A_{t'}$ and $t$ of the rest get together then they have $s_1, \ldots, s_{t'}$ from $A_1, \ldots, A_{t'}$, and the other $t$ can recover $s'$. Hence they can recover $s = s_1 \oplus \cdots s_{t'} \oplus s' = s$.

If some group gets together that does not have one of the $A_i$ for $1 \leq i \leq t'$ then they might get all the pieces EXCEPT $s'_i$, so they have NOTHING! If some group gets together that had $A_1, \ldots, A_{t'}$ and LESS THAN $t$ of the rest the don't have $s'$ so THEY HAVE NOTHING!

## References


[1] M. Bellare and P. Rogaway. Robust computational secret sharing and a unifed account of classical secret-sharing goals. In *ACM CCS '07: 14th ACM Conference on Computer and Communications Security*, 2007. `http://eprint.iacr.org/2006/449.pdf`.

[2] G. Blakely. Safeguarding cryptographic keys. In *Proceeding of the National Computer Conference*, pages 313–317, 1979.

[3] B. Chor, S. Goldwasser, S. Micali, and B. Awerbuch. Verifiable secret sharing and achieving simultaneity in the presence of faults. In *Proceedings of the 26th Annual IEEE Symposium on Foundations of Computer Science,* Portland OR, pages 383–395, 1985.

[4] P. Feldman. A practical scheme for non-interactive verifiable secret sharing. In *Proceedings of the 28th Annual IEEE Symposium on Foundations of Computer Science,* Los Angeles CA, pages 427–437, 1987.

[5] H. Krawdzyk. Secret sharing made short. In *Advances in Cryptology: Proceedings of CRYPTO '93,* Santa Barbara CA, 1993.

[6] A. Shamir. How to share a secret. *Communications of the ACM*, 22, 1979.