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

# 1 Introduction

The field of Secret Sharing was invented by Adi Shamir [**?**] and George Blakely [**?**] 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 **??** 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

$$f(x) = r_{t-1}x^{t-1} + \cdots + r_1x + s.$$

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 [**?**] scheme. Blakely [**?**] 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 **??** we saw how Zelda could share here secret with ANY set of subsets, though the schemes were far from ideal. In Section **??** 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

$$f(x) = r_{t-1}x^{t-1} + \cdots + r_1x + s'.$$

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!

# 7 Can We Use Shorter Shares?

If the secret is of length $n$ and any set of $t$ people can learn it, and everyone has the same length share, then everyone MUST have a share of length $n/t$. In the schemes above everyone had a share of length $n$. Can we do better? Can we do $n/t$? Can we come close?

## 7.1 Secret Sharing with shares of length $n/t$

**Theorem 7.1** *There is a secret sharing scheme where Zelda has a secret of length $n$, any $t$ people can learn the secret, no set of $t-1$ can learn anything about the secret, and everyone gets a share of length $n/t$.*

**Proof:** Zelda's secret is $s = s_0 s_1 s_2 \cdots s_{t-1}$ where each $s_i$ is of length $n/t$. Zelda finds a finite field $F$ such that $|F| = 2^{n/t}$. Zelda generates random $k$ of length $n$. $k = k_0 k_1 \cdots k_{t-1}$. Zelda uses the polynomails

$$f(x) = s_{t-1}x^{t-1} + \cdots s_1 x + s_0$$

For $1 \leq i \leq m$ Zelda gives $A_i$ $f(i)$.

KEY: Everyone gets a string of length $n/t$.

If $t$ of Zelda's associates get together then they can find $f$ and hence they know $k$ and can find $s$.

Each player gets a number in $GF(2^{n/t})$ hence of length $n/t$.

Why is this scheme secure?

YOU"VE BEEN PUNCKED! Lets look at what just $A_1$ knows. He knows

$$f(1) = s_{t-1} + \cdots + s_0.$$

That's already some information! In particular, $A_1$ can ELIMINATE some possibilities for the secret. $A_1, \ldots, A_{t-1}$ can eliminate even more. ▮

## 7.2 Secret Sharing with shares of length $2n/t$ Using 1-time Pad

The problem with the proof of Theorem **??** is that the actual secret is the coefficient. We fix this by having the coefficients be the secret coded with a 1-time pad. We do not get $n/t$ but we get $2n/t$.

**Theorem 7.2** *There is a secret sharing scheme where Zelda has a secret of length $n$, any $t$ people can learn the secret, no set of $t-1$ can learn anything about the secret, and everyone gets a share of length $2n/t$.*

**Proof:** Zelda's secret is $s = s_0 s_1 s_2 \cdots s_{t-1}$ where each $s_i$ is of length $n/t$. Zelda finds a finite field $F$ such that $|F| = 2^{n/t}$. Zelda generates random $k$ of length $n$. $k = k_0 k_1 \cdots k_{t-1}$. Zelda finds two polynomails

$$f(x) = (s_{t-1} \oplus k_{t-1})x^t + \cdots (s_1 \oplus k_1)x + (s_0 \oplus k_0)$$

$$g(x) = k_{t-1}x^t + \cdots k_1 x + k_0$$

For $1 \leq i \leq m$ Zelda gives $A_i$ $(f(i), g(i))$.

KEY: Everyone gets a string of length $2n/t$.

If $t$ of Zelda's associates get together then they can find both $f$ and $g$ hence they know $k$ and can find $s$.

Why is this scheme secure?

YOU"VE BEEN PUNKED! AGAIN! $t-1$ people CAN find out something about the secret. If $A_1, \ldots, A_{t-1}$ get together they can cycle through all $2^{2n/t}$ possible shares for $A_t$ and produce $2^{n/t}$ possibilities for the secret. Hence they can eliminate some possibliities. Not info-theoretic secure. ∎

Lets recap what we have so far and what to make of it.

Zelda has a secret $s$, with $|s| = n$. There are $m$ people. Zelda wants that if any $t$ get together they can recover the secret, but any $t-1$ cannot. The scheme in Section **??** does this! We point out two aspects of it:

- Any set of $t-1$ gets NO INFORMATION WHATSOEVER. They cannot even eliminate any possibility for the secret. We call the scheme information-theoretically secure.
- Every player gets from Zelda a string of length $n$.

Is there an information-theoretic secure scheme where some player get strings of length $n-1$? NO:

**Theorem 7.3** *Assume there are $m$ people $A_1, \ldots, A_m$ and a number $2 \leq t \leq m$. Assume there is an attempt at a threshold-t secret sharing scheme. If one of the players (we assume $A_t$ gets a string of length $n-1$ then the scheme is NOT information-theoretically secure.*

**Proof:** Assume there is a scheme scheme where $A_t$ gets a share of length $n-1$. If $A_1, \ldots, A_{t-1}$ get together they can try ALL of the $2^{n-1}$ possible shares that $A_t$ could have gotten and produce a set of $2^{n-1}$ possible secrets. Hence the $t-1$ players can eliminate HALF of the possibilites for secrets. This is NOT information-theoretically secure. ∎

## 7.3 Ramp Secret Sharing with Shorter Shares

By Theorem **??** we cannot have a scheme where some player gets a share of length less than $n$. So we lower our goals so that if $t-1$ people get together they can learn something, $t-2$ can learn less, and then at some point they can't learn at all.

**Theorem 7.4** *Let $1 \leq t' < t$. There is a secret sharing scheme such that the following hold.*

1. *If any $t$ people get together they can learn the secret.*
2. *If $\leq t'$ people get together they cannot learn the secret.*
3. *If $t' \leq L \leq t$ people get together then they can narrow the secret down to a space of size $(2^{n/g})^{t-L} = 2^{(n/t-t')(t-L)}$.*
4. *Every player gets a share of size $n/(t-t')$.*

**Proof:** Let $g$ be a parameter to be determined later. Zelda's secret is $s = s_1 s_2 s_3 \cdots s_g$ where each $s_i$ is of length $n/g$. Zelda finds a finite field $F$ such that $|F| = 2^{n/g}$. Zelda finds a polynomial $p$ of degree $t-1$ such that for $0 \leq i \leq g-1$, $f(i) = s_i$. For $1 \leq i \leq m$ Zelda gives $A_i$ $f(g+i)$.

KEY: Everyone gets a string of length $n/g$.

If $t$ of Zelda's associates get together then they can interpolate and find the polynomial $f$. We need to find $g$. If $t'$ people get together they can cycle through the shares that the $t-t'$ people would get to reduce the number of possible

$$(2^{n/g})^{t-t'} \geq 2^n$$

$$2^{n(t-t')/g} \geq 2^n$$

$$n(t-t')/g \geq n$$

$$(t - t')/g \geq 1$$

$$t - t' \geq g$$

$$g \leq t - t'.$$

We can take $g = t - t'$.

NOW- if $t' \leq L \leq t$ people get together then they can cycle through the shares that $t - L$ people would use to reduce the number of possibilities to $(2^{n/g})^{t-L} = 2^{(n/t-t')(t-L)}$.

Can they learn more than this? No, but I won't prove that. This time **YOU HAVE NOT BEEN PUNKED!** ▮

This is called *Ramp Secret Sharing* since we picture a ramp betwee $t'$ and $t$. As the number of people goes from $t'$ to $t$ they can learn more and more.

### 7.4 Secret Sharing with Shorter Shares

By Theorem **??** we CANNOT do better if we want info-theoretic security. One way around this problem is Ramp Secret Sharing, as shown in Theorem **??**. Another way is to weaken the security from info-theoretic to computational. Our exposition is essentially that of Hugo Krawczyk [**?**] with one difference we will discuss when we talk about the security.

We give three approaches. The first one does not work but gives us ideas for the next one. The second one gives a reduction in size.

Recall that there are $m$ people and the secret $s$. We want that if $t$ of them get together they can find it.

### 7.5 Shorter Shares: Split up the Secret

Instead of using the $s$ as the constant for a degree $t - 1$ polynomial we will split $s$ into $t$ pieces and use each piece as a coefficient.

1. Let $s = s_0 s_1 \cdots s_{t-1}$ where all of the $s_i$'s are of roughly the same length. We assume $|s_i| = |s|/t$.
2. Let $F = GF(2^{|s|/t})$.
3. Zelda forms the polynomial (over $F$)

$$f(x) = s_{t-1}x^{t-1} + s_{t-2}x^{t-2} + \cdots + s_1 x + s_0.$$

4. All arithmetic is mod $p$. Zelda gives $A_i$ the number $f(i)$.

Each player gets a number in $\{0, \ldots, p-1\}$, hence of length at most $\lg p \leq \lg 2s^{1/t} = \lg(s)/t + 1 = \frac{|s|}{t} + 1$.

If $t$ of them get together they can recover the polynomial and hence the secret.

If $t-1$ of them get together then what do they know? Well, lets look at what just $A_1$ knows. He knows

$$f(1) = s_{t-1} + \cdots + s_0 \pmod{p}.$$

That's already some information! If two people get together they can do some algebra and learn some more. They won't learn the secret but they will learn things about the secret. Hence this scheme is not information-theoretically secure. Is it computational secure? I doubt it, but I doubt its known since I invented this scheme for this course and its not well studied.

In the next section we give a scheme that is computationally secure.

### 7.6 Shorter Shares: Split up the Shares and use RSA

**Def 7.5** If Alice and Bob use RSA with primes $p, q$ and encryption key $e$, decryption key $d$ then $RSA(p, q, e, d, m)$ is what Bob sends Alice if he wants to send $m$. If $p, q, e, d$ are understood (this will always be the case) then we may just use $RSA(m)$. (We use $RSA$ but any public key crypto system would work in this section.)

As in the last Section the secret is split up. However, before forming the polynomial Zelda encodes the $s_i$ with RSA. She will also (1) broadcast to everyone $n, e$, and (2) give everyone a second polynomial which codes $p, d$.

Recall that Zelda wants to share a secret $s$ with such that if $t$ people get together they can find it out, but if $t-1$ people get together they cannot. The number of people will not matter.

1. Zelda picks a $p, q, e, d$ that (1) satisfy the conditions of RSA, and (2) $p$ and $q$ are roughly $2^{|s|/t}$, so $|p| = |q| = |s|/t + O(1)$ and $n = pq$ is such that $|n| = |s|/t + O(1)$. Henceforth we ignore $O(1)$ terms, so we take $|p| = |q| = |n| = |s|/t$.

2. Zelda computes $u = RSA(s)$. We assume $|u| = |s|$.

3. Zelda takes $u = u_0 \cdots u_{t-1}$ where all of the $u_i$'s are of roughly the same length. We take $|u_i| = |s|/t$.

4. Let $F = GF(2^{|s|/t})$. Note that all $u_i$ are in $F$.

5. Zelda forms the polynomial (over $F$)

$$f(x) = u_{t-1}x^{t-1} + u_{t-2}x^{t-2} + \cdots + u_1 x + u_0.$$

6. Let $k = (p, d)$ be a way to code $p$ and $d$ into one number. We can arrange things such that $|(p, d)| = 2|s|/t$.

   We use a field $F'$ on $2^{2|s|/t}$ elements. Zelda will ALSO secret-share the key $k$. This we do in the standard way; however we still describe it for completeness and so we can our analysis.

   Zelda picks random numbers $r_{t-1}, \ldots, r_1 \in F$ (so $|r_i| \leq |s|/t$). Zelda forms the polynomial (over $F'$)

$$g(x) = r_{t-1}x^{t-1} + r_{t-2}x^{t-2} + \cdots + r_1 x + k.$$

7. Zelda gives $A_i$ the numbers $f(i)$ and $g(i)$. Zelda also gives everyone $(n, e)$ but we won't count that as a share since everyone gets it.

   How many bits does Zelda give each $A_i$?

   $f(i)$ is of length $|s|/t$.

   $g(i)$ is of length $2|s|/t$.

   Hence the total length is $3|s|/t$.

If $t$ of them get together they can recover the polynomial $f(x)$ and the polynomial $g(x)$ so they will have the $u_0, \ldots, u_{t-1}$ and $k = (p, d)$. They can thus find $u = u_0 \cdots u_{t-1}$ and the factorization of $n$, hence they can compute $s = RSA^{-1}(u)$.

If $t-1$ of them get together then what do they know? For ten years this scheme was thought to be secure. Then it was broken by Bellare and Rogaway [**?**]. Actually, thats not quite fair. The original protocol *did not* specify RSA. It just said to use *some* public key encryption. Bellare and Rogaway found that *it matters* which one is used. RSA does not have the properties needed. But other public key encryption schemes do. So it is more fair to say that Bellare and Rogaway clarified and corrected Krawczyk's construction. But that is also not quite fair. Bellare and Rogaway did more than that- they put Secret Sharing on a more rigorour footing.

Can we do better than $2|s|/t$?

We can! (and this time I am not punking you).

We can't beat $|s|/t$. Lets view the last $2|s|/t$ result as

$$|s|/t(\text{ share of encoded secret }) + |s|/t(\text{ share of Key }).$$

Can we decrease how much we need for the share of the Key. We can! But first we will express the shares for the encoded secret differently.

1. Zelda picks a $p, q, e, d$ that (1) satisfy the conditions of RSA, and (2) $p$ and $q$ are roughly $2^{|s|/2t}$, so $|p| = |q| = |s|/2t + O(1)$ and $n = pq$ is such that $|n| = |s|/2t + O(1)$. Henceforth we ignore $O(1)$ terms, so we take $|p| = |q| = |n| = |s|/2t$.

2. Zelda computes $u = RSA(s)$. We assume $|u| = |s|$.

3. (NEW STEP) Zelda takes $u = u_{1,0} \cdots u_{1,t-1} u_{2,0} \cdots u_{2,t-1}$. where all of the $u_i$'s are of roughly the same length. We take $|u_i| = |s|/2t$. So the $u_i$'s are all shorter!

4. Let $F = GF(2^{|s|/2t})$. Note that all $u_i$ are in $F$.

5. Zelda forms TWO polynomials (over $F$)

$$f_1(x) = u_{1,t-1}x^{t-1} + u_{1,t-2}x^{t-2} + \cdots + u_{1,1}x + u_{1,0}.$$

$$f_2(x) = u_{2,t-1}x^{t-1} + u_{2,t-2}x^{t-2} + \cdots + u_{2,1}x + u_{2,0}.$$

6. Let $k = (p, d)$ be a way to code $p$ and $d$ into one number. We can arrange things such that $|(p, d)| = |s|/t$. Zelda will ALSO secret-share the key $k$. This we do in the standard way; however we still describe it for completeness and so we can our analysis. We use a field $F'$ on $2^{2|s|/2t} = 2^{|s|/t|}$ elements.

   Zelda picks random numbers $r_{t-1}, \ldots, r_1 \in F$ (so $|r_i| \leq |s|/2t$). Zelda forms the polynomial (over $F'$)

$$g(x) = r_{t-1}x^{t-1} + r_{t-2}x^{t-2} + \cdots + r_1 x + k.$$

7. Zelda gives $A_i$ the numbers $f_1(i), f_2(i)$ and $g(i)$. Zelda also gives everyone $(n, e)$ but we won't count that as a share since everyone gets it.

   How many bits does Zelda give each $A_i$?

   $f_1(i)$ is of length $|s|/2t$.

   $f_2(i)$ is of length $|s|/2t$.

   $g(i)$ is of length $|s|/t$.

   Hence the total length is

$$2\frac{|s|}{2t} + \frac{s}{t} = \frac{|s|}{t} + \frac{|s|}{2t}$$

   I kind of like this form since it is clearly THE SECRET plus THE KEY YOU NEED TO GET THE SECRET. Even so, we have to add them up to get

$$\frac{3|s|}{2t}$$

   The number of bits needed to share the $u_i$'s did not change- view it as $2 \times |s|/2t$ instead of $|s|/t$, but its still the same. The number of bits needed to share the key has now gone down.

   One could iterate this method further.

# 8 Verifiable Secret Sharing

Verifiable secret sharing was introduced by Chor, Goldwasser, Micali, Awerbuch [**?**]. We give a scheme due to Feldman [**?**].

Zelda and $A_1, \ldots, A_9$ are the players. Zelda wants any 5 of them to be able to crack the secret but no 4 of them. So they will use the poly method.

1. The secret is $s$. $p$ is chosen such that $s \leq p \leq 2s$. Zelda picks random $r_4, r_3, r_2, r_1 d$ and form the polynomial $f(x) = r_4x^4 + r_3x^3 + r_2x^2 + r_1x + s$.

2. For $1 \leq i \leq 9$ Zelda gives $A_i$ the element $f(i)$.

$A_2, A_4, A_7, A_8, A_9$ get together. BUT $A_7$ DOES NOT WANT the group to find the secret out! Hence we need to VERIFY that everyone is telling the truth. This is called VERIFIABLE secret sharing.

We give a scheme that is computationally secure, but not information-theoretic secure. Aside from Zelda giving the players the above she also does the following.

1. Zelda finds a generator $g$ for the prime $p$.

2. Zelda gives to EVERYONE the values $g^{r_1}$, $g^{r_2}$, $g^{r_3}$, $g^{r_4}$, $g^{r_5}$, and $g^s$ AND $g$ itself. (Recall- we think Discrete Log is HARD so this information does not reveal $r_1, r_2, r_3, r_4, r_5$ or $s$.)

NOW when $A_2, A_4, A_7, A_8, A_9$ get together they do the following:

1. $A_2$ reveals what $f(2)$ is. Let what $A_2$ reveals be called $f2q$ (stands for $f(2)$?). We will be able to TEST if it really is $f(2)$.

2. They all compute the following to VERIFY that $f(2) = f2q$.

$$(g^s)^{2^0} \times (g_1^r)^{2^1} \times (g_2^r)^{2^2} \times (g_3^r)^{2^3} \times (g_4^r)^{2^4} = g^s \times g^{2^1 r_1} \times g^{2^2 r_2} \times g^{2^3 r_3} \times g^{2^4 r_4} = g^{s+2^1 r_1+2^2 r_2+2^3 r_3+2^4 r_4} = g^{f(2)}$$

SO, the result is REALLY $g^{f(2)}$. NOW they all compute $g^{f2q}$. If they MATCH then this VERIFIES that $f2q = f(2)$ without revealing anything else about the polynomial.

3. We do similar things to verify the other $f$'s.

4. Once all of the $f$'s are verified they can use them to reveal what the secret is.

Verifiable Secret Sharing has the advantage that if someone is lying you can detect that someone is lying, though you won't know who. However, there is a disadvantage: poly-secret sharing and the random-string secret sharing are all information-theoretic secure. Verifiable secret sharing is only secure if none of the players can compute discrete log.

The VSS scheme above has one other advantage. Lets say $t' \geq t$ people get together of which $t$ are honest. Then the secret can be found: first find out who is lying and do not use them.