Spring 2023 - ICSI 526
Homework 3

Written by Jacob Clouse

Partner 1: Jacob Clouse with Shares: (37,9) and (37,18)
Partner 2: Luna Dagci with Shares: (38,24) and (38,2)

April 17, 2023

Contents

1 Question 1

1.1 Trying to find secret values

I am Partner 1 in this instance and my shares are: (37,9) and (37,18), with our prime being 31. These are two different shares from two different equations, and we know that the original cipher is a (2, n) SSS scheme. This means we need 2 shares (k) at least in order to retrieve it, but we have only 1 for each (k - 1). We know that from our text that we can not find the original value from this, but lets look deeper. To retrieve our original secret (a_0), we use the formula:

$$f(37) = (a_0 + a_1 * 37) \bmod 31$$

Lets use (37,9) to try and fill in as much as we can with the formula. Lets say we have $a_0 = 3$ and $a_1 = 1$, then we have:

$$f(37) = (3 + 1 * 37) \bmod 31 \rightarrow 9$$

Or lets say we have $a_0 = 34$ and $a_1 = 1$, then we have:

$$f(37) = (34 + 1 * 37) \bmod 31 \rightarrow 9$$

Or lets say we have $a_0 = 59$ and $a_1 = 2$, then we have:

$$f(37) = (59 + 2 * 37) \bmod 31 \rightarrow 9$$

How do we know which one is correct? They all give us the same answer and each one is equally likely. This is why we can't know for sure and is why it is impossible to find the secret without having k shares.

1.2 Finding Average

For the average, I had to take my two shares and average the y coordinates together, my partner had to do the same with their shares and then we could feed the result into the Lagrange Formula to get our f(0) value.
Again, as Partner 1, my two shares were: (37,9) and (37,18). So I initially thought I had to just average 9 and 18, but that will lead to a decimal. So i had to use the forumla: (9+18)*2^{-1} mod 31. Here is **my** math:

$$(9+18)*2^{-1} \bmod 31 \rightarrow (27)*16 \bmod 31 \rightarrow 432 \bmod 31 \rightarrow \mathbf{29}$$

Luna was Partner 2 and her two shares were: (38,24) and (38,2). She had to average 24 and 2, here is **her** math:

$$\left(\tfrac{24+2}{2}\right) \bmod 31 \rightarrow \left(\tfrac{26}{2}\right) \bmod 31 \rightarrow (13) \bmod 31 \rightarrow \mathbf{13}$$

We now had our two averaged shares: (37,29) and (38,13). We plugged them into our Lagrange Formula and got the following:

$$f(x) = y_1 * \left(\tfrac{x-x_2}{x_1-x_2}\right) + y_2 * \left(\tfrac{x-x_1}{x_2-x_1}\right) \bmod 31 \rightarrow f(x) = 29 * \left(\tfrac{x-38}{37-38}\right) + 13 * \left(\tfrac{x-37}{38-37}\right) \bmod 31$$

$$f(0) = 29 * \left(\tfrac{0-38}{37-38}\right) + 13 * \left(\tfrac{0-37}{38-37}\right) \bmod 31 \rightarrow f(0) = 29 * \left(\tfrac{-38}{-1}\right) + 13 * \left(\tfrac{-37}{1}\right) \bmod 31$$

$$f(0) = 29*38 + 13*(-37) \bmod 31 \rightarrow f(0) = 1102 + (-481) \bmod 31 \rightarrow f(0) = 621 \bmod 31 \rightarrow f(0) = 1$$

f(0) = 1 for the averaged formula, we just need to work backwards and see if we can get the original values. We find secret 1 with shares (37,9) and (38,24):

$$f(x) = 9 * \left(\tfrac{x-38}{37-38}\right) + 24 * \left(\tfrac{x-37}{38-37}\right) \bmod 31$$

$$f(0) = 9 * \left(\tfrac{0-38}{37-38}\right) + 24 * \left(\tfrac{0-37}{38-37}\right) \bmod 31 \rightarrow f(0) = 9 * \left(\tfrac{-38}{-1}\right) + 24 * \left(\tfrac{-37}{1}\right) \bmod 31$$

$$f(0) = 9*38 + 24*(-37) \bmod 31 \rightarrow f(0) = 342 + (-888) \bmod 31 \rightarrow f(0) = -546 \bmod 31 \rightarrow f(0) = 12$$

f(0) for secret 1 is 12. We find secret 2 with shares (37,18) and (38,2):

$$f(x) = 18 * \left(\tfrac{x-38}{37-38}\right) + 2 * \left(\tfrac{x-37}{38-37}\right) \bmod 31$$

$$f(0) = 18 * \left(\tfrac{0-38}{37-38}\right) + 2 * \left(\tfrac{0-37}{38-37}\right) \bmod 31 \rightarrow f(0) = 18 * \left(\tfrac{-38}{-1}\right) + 2 * \left(\tfrac{-37}{1}\right) \bmod 31$$

$$f(0) = 18*38 + 2*(-37) \bmod 31 \rightarrow f(0) = 684 + (-74) \bmod 31 \rightarrow f(0) = 610 \bmod 31 \rightarrow f(0) = 21$$

f(0) for secret 2 is 21. Now we take these numbers and can feed them back into our average formula: (12+21)*2^{-1} mod 31. Here is **my** math:

$(12 + 21) * 2^{-1} \bmod 31 \rightarrow (33) * 16 \bmod 31 \rightarrow 526 \bmod 31 \rightarrow \mathbf{1}$

$1 = 1$, but why? The reason lies with the operations that we performed on the data: addition and scalar division.
We know that both addition and scalar division are supported homomorphic operations inside of Shamir's Secret Sharing sharing. They

1.3 Finding Multiplication

To calculate the multiplication value, I had to multiply my two share y values and then find mod 31 of the product. My partner had to do the same with their shares and then we could feed the result into the Lagrange Formula to get our f(0) value.
Again, as Partner 1, my two shares were: (37,9) and (37,18). So I had to multiply 9 and 18, here is **my** math:

$(9 * 18) \bmod 31 \rightarrow (162) \bmod 31 \rightarrow \mathbf{7}$

Luna was Partner 2 and her two shares were: (38,24) and (38,2). She had to multiply 24 and 2, here is **her** math:

$(24 * 2) \bmod 31 \rightarrow (48) \bmod 31 \rightarrow \mathbf{17}$

We now had our two multiplied shares: (37,7) and (38,17). We plugged them into our Lagrange Formula and got the following:

$$f(x) = y_1 * \left(\frac{x-x_2}{x_1-x_2}\right) + y_2 * \left(\frac{x-x_1}{x_2-x_1}\right) \bmod 31 \rightarrow f(x) = 7 * \left(\frac{x-38}{37-38}\right) + 17 * \left(\frac{x-37}{38-37}\right) \bmod 31$$

Solving for f(0):

$$f(0) = 7 * \left(\frac{0-38}{37-38}\right) + 17 * \left(\frac{0-37}{38-37}\right) \bmod 31 \rightarrow f(0) = 7 * \left(\frac{-38}{-1}\right) + 17 * \left(\frac{-37}{1}\right) \bmod 31$$

$$f(0) = 7 * 38 + 17 * (-37) \bmod 31 \rightarrow f(0) = 266 + (-629) \bmod 31 \rightarrow f(0) = -363 \bmod 31 \rightarrow f(0) = 9$$

f(0) = **9** for the multiplication formula, we just need to work backwards and see if we can get the original values. We from 1.2 that secret 1 = 12 and secret 2 = 21. But we see that when we plug in these values into the original formula, we are left with a problem:

$(12 * 21) \bmod 31 \rightarrow 252 \bmod 31 \rightarrow 4$

The reason is because full, non-scalar multiplication is NOT supported within Shamir's Secret Sharing. Scalar multiplication would have been supported (scalar multiplication is multiplying by a constant). Because we are multiplying each share against each other (and NOT a constant), this is full multiplication and

it will work.

2 Question 2

2.1 How to Run My Shamir Secret Sharing program (Non-Downscaling)

This will encrypt and decrypt the input image WITHOUT changing the dimensions of the output. It is hard coded to make a (2,5) cipher with 5 total shares and 2 required shares to reconstruct.
Instructions:

1. Make sure you have the numPy, openCV, PIL and random libraries installed and accessible in your environment.
2. Make sure you have a SQUARE sample bitmap in the same location as the **Jacob_Clouse_Q2_SSS.py** file. Both length and width must be the same for it to work (for example, the picture can be 640 x 640).
3. You can run the file by entering the file's directory in a terminal and running: **python Jacob_Clouse_Q2_SSS.py** (if you have python2 and python3 installed, you might need to use: **python3 Jacob_Clouse_Q2_SSS.py**)
4. After my logo, you will be asked to input the path/name of the input bitmap WITH the .bmp extension (if you have placed the image file in same folder, you can enter in just the name like so: **example.bmp**)
5. It will then ask you if you want to downsize, enter in 'no' (as we want to preserve our original dimensions).
6. You will see it create the 5 shares in the same directory and then it will output file (it will have 'reconstructed' in its name and the dimensions of the image). You are all done!

We take the image and open it up in our first, then adjust all values so that the max is 250. Once we get our pixel values, we then generate our coefficients and create our shares. We then feed the shares into our decrypt function and get our reconstructed image back.

2.2 How to Run My Shamir Secret Sharing program (Downscaling)

This will encrypt and decrypt the input image WIT changing the dimensions of the output to be half width and half height of the original. It is hard coded to make a (2,5) cipher with 5 total shares and 2 required shares to reconstruct.
Instructions:

1. Again, Make sure you have the numPy, openCV, PIL and random libraries installed and accessible in your environment.
2. Again, Make sure you have a SQUARE sample bitmap in the same location as the **Jacob_Clouse_Q2_SSS.py** file. Both length and width must be the same for it to work (for example, the picture can be 640 x 640).
3. Again, You can run the file by entering the file's directory in a terminal and running: **python Jacob_Clouse_Q2_SSS.py** (if you have python2 and python3 installed, you might need to use: **python3 Jacob_Clouse_Q2_SSS.py**)

4. After my logo, you will be asked to input the path/name of the input bitmap WITH the .bmp extension (if you have placed the image file in same folder, you can enter in just the name like so: **example.bmp**)

5. It will then ask you if you want to downsize, enter in 'yes' (the only difference from non-downscaling, we want to downsize the width and height to be half of the original values).

6. You will see it create the 5 shares in the same directory and then it will output file (it will have 'reconstructed' in its name and the dimensions of the image, which should be half of the original).

   The file will be downscaled multiple times, once it will be downscaled directly before encryption of the shares.

   The second time we downscale will be the shares after encryption. After they are generated, we take each and downscale them accordingly.

7. The reconstruction will be the same and you will be able to view the MAE in the console window. After this, you are all set!