Public key encryption Scheme for large image

Vendula Maulerova, 921031-7161

FYST13

Chaos in science and technology

Lund University

Sweden

1 Introduction to ECC cryptography

1.1 Introduction

Computer cryptography uses advantage of the problems that are difficult to solve, but easy to verify: an example can be multiplication by two very big prime numbers $m = ab$. Knowing m does not make it easy to find a and b, however knowing a, b is easy to calculate by a simple division $b = m/b$. In general the question whether all problems that are verifiable in polynomial time are also solvable in polynomial time remains unknown and has been classified as one of the seven Millenium Prize Problems[1]. Modern cryptography has its roots in using the number theory to generate private keys that are used for decryption and public keys that are used as a lock that can transfer the secret.

1.2 RSA and Diffie Hellman

Classic RSA [2, 3] generates the encryption in a following way: Alice and Bob share and arbitary number G. Alice has a secret key k_A, Bob has a secret key k_B. Alice produces number $H_A = G^{k_A}$ and Bob produces number $H_B = G^{k_B}$. Those number are exchange and both users again raise their exponent on their secret keys, such that Alice has $S = H_B^{k_A} = G^{k_B k_A}$ and Bob has $S = H_A^{k_B} = G^{k_A k_B}$ where G, H_B and H_C are public key. Diffie Hellman complicates the algorithm by adding modulo operator in a following way: Alice produces number $H_A = G^{k_A}$ mod (p) and Bob produces number $H_B = G^{k_B}$ mod (p) where p is a prime number. This algorithms had been secure way for exchanging information, however nowadays as the computer power increases, the encryption algorithms must take longer times. Also a need for quantum-hard encryption algorithms is arising as the quantum computers are slowly coming from a conceptual stage to the first prototypes [4, 5, 6].

1.3 ECC cryptography

The alternative to a classic RSA Diffie-Hellman key exchange is an Elliptic curve cryptography. Elliptic curve is defined as:

$$E(F_p) = (x, y)|y^2 \equiv x^3 + ax + b(modp), x, y \in F_p \cup 0 \tag{1}$$

where $F_p = 0, 1, ..., p-1$ and p is a large prime number and $a, b \in F_p$, s.t. $4a^3 + 27b^2 \not\equiv 0(modp)$. The Elliptic curve cryptography require much shorter keys in order to achieve the same security level as RSA Diffie-Hellman [7]. An intuition for an elliptic curve can be developed by looking at the continuous (non-discrete) version of the equation 1 which can be seen in Fig. 1. The

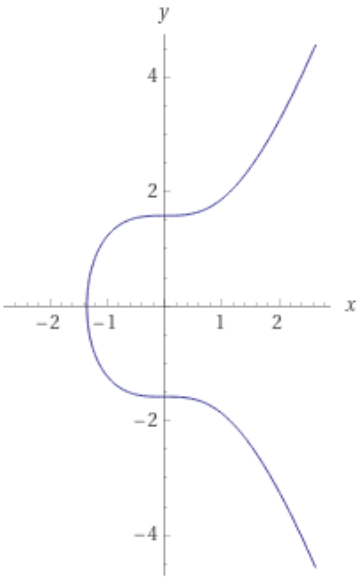


Figure 1: The elliptic curve corresponding to the equation $y^2 \equiv x^3 + 2.5$. Figure generated by Wolfram Alpha [8].

equivalent of key generation for Alice is Alice generates point $P = (x_p, y_p)$ as a generator point on certain curve and multiply that by k_A. Multiplication however in this case means, moving the point on an elliptic curve k_A. Pycryptodome currently supports implementation of 20 different curves, in the project, curve and generator point by 'NIST p-256', the description of the curve can be found in NIST database of Digital Signature Standards [9] (in the algorithm used in this project, the ECC classes are taken from pycryptodome, the moving of the point and subsequently multiplication by gmpInteger are redifined in those classes). Bob obtains numbers $P, Q = k_A * P$ where k_A remains Alice's secret and the type of curve she used. He encrypts the picture by adding two secret keys k_B, k_C such that $K_0 = k_B Q = k_B k_A P$ and $L = k_B Q = k_c k_A P$, to which he generates a matrix and subsequently uses snake addition and Arnold chaotic

map for actual encryption of the image. Alice can then generate the same initial matrix and by applying reverse operations can see the original image without the interception of a third party.

2 Arnold cat map and chaos

The code uses a snake addition, which is defining the next pixel in the picture grid as a sum of itself and the previous one. Furthermore on several places it uses the modified Arnold cat map. The standard Arnold's cat map is defined as follows:

$$x(j+1) = x(j) + y(j)y(j+1) = x(j) + 2y(j)$$

. The Jacobian of this system is:

$$D_f = \begin{pmatrix} 1 & 1 \\ 1 & 2 \end{pmatrix} \tag{2}$$

with the eigenvalues $H_{1,2} = (3 \pm \sqrt{5})/2$ and corresponding Lyapunov exponents $\lambda_{1,2} = \pm 0.962$. This implementation of the code however uses modified Arnold map with the Jacobian as follows:

$$D_f = \begin{pmatrix} 1 & c \\ d & cd+1 \end{pmatrix} \tag{3}$$

The eigenvalues of the modified Arnold map can be calculated as follows:

$$\begin{vmatrix} 1-h & c \\ d & cd+1-h \end{vmatrix} = 0 \tag{4}$$

, from where the eigenvalues can be expressed as:

$$h_{1,2} = \frac{1}{2}(2 + cd \pm \sqrt{c^2d^2 + 4cd}) \tag{5}$$

. The Lyapunov exponents are then:

$$\lambda_1 = ln(\frac{1}{2}(2 + cd + \sqrt{c^2d^2 + 4cd}))$$
$$\lambda_1 = ln(\frac{1}{2}(2 + cd - \sqrt{c^2d^2 + 4cd}))$$

Since $c > 0$ and $d > 0$ the Lyapunov exponent is always positive and this is a chaotic map. The complete definition for the Modified Arnold map procedure in the code is expecting parameters c, d, s, e, f and additionally takes values e, f and adds to the current pixel:

$$\begin{bmatrix} x^{'} \\ y^{'} \end{bmatrix} = \begin{bmatrix} 1 & c \\ d & cd+1 \end{bmatrix}^{s} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} e \\ f \end{bmatrix} (modN) \tag{6}$$

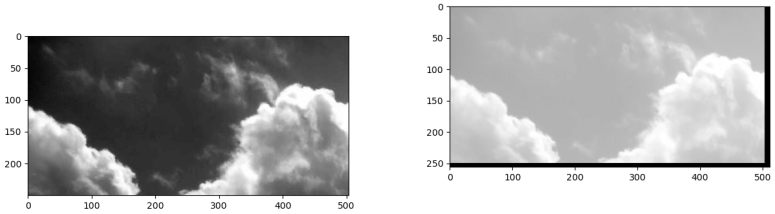
.

3 Code implementation and encrypted images

git clone https://github.com/Vendusha/ElipticCurveCryptography.git

Prerequisities: are csv library and pycryptodome. The older version of pycryptodome does not support Eliptic Curve Cryptography classes and the csv library requires python¡ 3.8. Recommended installation is to use python 3.7 and the following: conda install matplotlib numpy
conda install -c anaconda opencv
conda install -c conda-forge pycryptodome
The code has been implemented according to [10]. Here, the outline of the algorithm is described together with the test photo illustrations. Alice generates the secret key, that is stored in the file "AlicePrivateKey.pem". A public key is sent to Bob. Bob in between loads an image. Since the encryption code is run on personal computer, all images are in greyscale to save computational time. An example of an input image is shown in figure 2a. The Arnold mapping requires the inputs to be n × n, therefore Bob needs to do preprocessing on the image. Firstly if the image dimensions are not divisible by 256, the picture is filled with black pixels to have new dimensions (Fig. 2b). Secondly, image is divided to a number of images, each with dimensions 256x256 (Fig. 3). Function for preprocessing and merging the image back is called several times during the algorithm (the snake encryption is always called on the whole picture, while the

arnold modified mapping is called on each 256 x 256 picture separately.)



(a) Example of an input image

(b) Input Image is having a black stripes in x and y directions to have dimensions divisible by 256.

Figure 2: Original and preprocessed image.

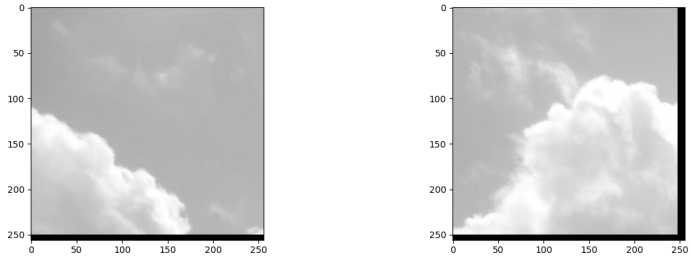


Figure 3: The image is converted to an image array of 2 images 256 x 256 pixels.

After preprocessing, Bob takes Alice's public key and generates matrix of keys. After the matrix is generated he does the encryption on the images, which contains the main steps as follows:

- Addition: he adds the array of matrices generated with his private key.
- Snake addition: Bob merges the images into one and encrypts pixel x(n+1)=x(n)+x(n+1) recursively.
- Bob splits the image into several 256 $\times$ 256 and applies Modified Arnold Map as described by Eq. 6.
- Snake addition II: Bob again merges the images into one and applies snake addition.

- Bob performs "exclusive or" bitwise operation on the key matrices and the encrypted image pixel by pixel.
- Lastly Bob sends Alice the encrypted image (see Fig. 4) together with the numbers $R = k_B P$ and $S = k_C P$ where k_B, k_C are his secret keys and P is the same generator point that Alice gave him (again, here the multiplication does not mean multiplication in the algebraic sense, but rather moving the point k_B resp. k_C times on elliptic curve). With this information, Alice can now decrypt the image.

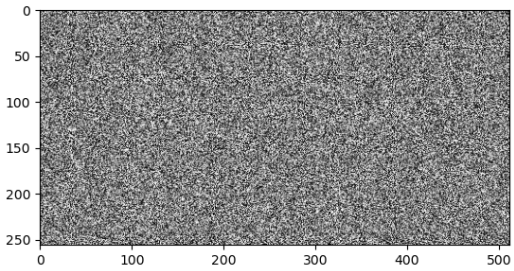


Figure 4: Example of encrypted image.

Now, with the knowledge of encrypted image C, and numbers R and S, Alice can decrypt the image:

- Recovery of key matrix: by performing operation $k_A * R$ and $k_A * S$ she recovers the key matrix that Bob made before.
- Alice performs "exclusive or" operation bitwise on the key matrices and encrypted image pixels.
- Snake decryption: Alice merges all the images together and does a reverse snake addition.
- Arnold-map decryption: Alice splits all the images and applies inverse procedure (using inverse matrix) to decrypt Arnold-map.

- Snake decryption II: Alice merges all the images again and applies inverse snake addition.
- Alice splits the images and subtract the matrix of keys.
- By merging the images again, Alice recovered the full image as seen in picture 5.

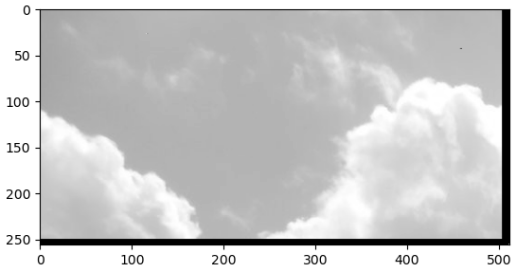


Figure 5: Image after decryption.

References

[1] Stephen Cook. The p versus np problem. In *Clay Mathematical Institute; The Millennium Prize Problem*, 2000.

[2] W. Diffie and M. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, 22(6):644–654, 1976.

[3] Ronald L Rivest, Adi Shamir, and Len Adleman. On digital signatures and public-key cryptosystems. Technical report, MASSACHUSETTS INST OF TECH CAMBRIDGE LAB FOR COMPUTER SCIENCE, 1977.

[4] T. M. Fernández-Caramès and P. Fraga-Lamas. Towards post-quantum blockchain: A review on blockchain cryptography resistant to quantum computing attacks. *IEEE Access*, 8:21091–21116, 2020.

[5] John Preskill. Quantum computing in the nisq era and beyond. *Quantum*, 2:79, 2018.

[6] Nikodem Grzesiak, Reinhold Blümel, Kenneth Wright, Kristin M. Beck, Neal C. Pisenti, Ming Li, Vandiver Chaplin, Jason M. Amini, Shantanu Debnath, Jwo-Sy Chen, and Yunseong Nam. Efficient arbitrary simultaneously entangling gates on a trapped-ion quantum computer. *Nature Communications*, 11(1):2963, Jun 2020.

[7] Thomas Wollinger, Jan Pelzl, Volker Wittelsberger, Christof Paar, Gökay Saldamli, and Çetin K Koç. Elliptic and hyperelliptic curves on embedded μp. *ACM Transactions on Embedded Computing Systems (TECS)*, 3(3):509–533, 2004.

[8] Wolfram—Alpha. Tree. From MathWorld—A Wolfram Web Resource. Last visited on 20/10/2020.

[9] Digital signature standard (DSS). Technical report, July 2013.

[10] L. Chen, X. Chen, and Z. Peng. A novel public key encryption scheme for large image. In *2014 IEEE 13th International Conference on Trust, Security and Privacy in Computing and Communications*, pages 955–960, 2014.