ISSN 2047-3338

Image crypto-compression based on Elliptic Curves and Linear Feedback Shift Register (LFSR)

Cidjeu Djeuthie Diderot and Tieudjo Daniel

***Abstract*—Images are represented in several forms to facilitate their processing and security. These last years, elliptic curves have demonstrated remarkable performances in Cryptography. For Elliptic Curve Cryptography (ECC) to be applied on images, they should be represented as points on elliptic curves. in this paper, we present a formalism to transform an image into a sequence of points of an elliptic curve. A stream encryption based on the Linear Feedback Shift Register (LFSR) is proposed and used to encrypt images seen as sequences of points on elliptic curves. The JPEG compression is combined to this encryption to produce a new crypto-compression scheme. This scheme is implemented on sample images. The obtained results, compared to existing works, offer better satisfaction in terms of integrity.**

***Index Terms*—crypto-compression, image, LFSR and Elliptic Curves**

I. Introduction

SEVERAL domains of the digital world rely on images: road safety, telemedicine, video conferences, etc. These images often contain sensitive data. In telemedicine for example, data are inserted into images to identify a patient, his medical examination, his treatment, etc. Because such images are digital files which can be transferred through public channels, they need to be secured [15].

In recent years, image security has been the subject of several research works ([1], [18], [25]). Existing works rely on cryptographic tools such as AES ([4], [9], [17], [21]), Chaos Theory ([10], [24], [25]), stream/bloc encryption ([19], [2]), etc. To practically secure images, compression is joined to encryption to obtain a hybrid process called crypto-compression. In 2004, Ftérich and al. proposed a crypto-compression system based on AES encryption and quadtree compression [9]. In 2006, Puech and al. also used AES encryption for crypto-compression based on Discrete Cosine Transform (DCT) [21]. Puech and al's crypto-compression system offers better compression ratio due to DCT. However, DCT produces visible losses at a certain level of compression. A solution to this problem has been proposed by Benabdellah and al., who presented two crypto-compression algorithms based on the Faber-Schauder Multi-Scale Transform (FMT). The first algorithm combines FMT and DES [3] while the second puts together FMT and AES [4]. Although these two schemes offer better quality-compression ratio, they need a secret key which is generally shared through symmetric encryption and key exchange protocols. These protocols are built on problems of Number Theory based Cryptography and are heavy when considering keys size and operations. More recently, Chaos-based Cryptography has been used for image crypto-compression. In [11], chaos-based encryption is joined to DCT compression, when in [16], chaos-based pseudorandom bit generator is combined to arithmetic coding (AC) compression. When evaluating these previous works, it appears that there is a compromise between the compression ratio, the execution time and the quality of the image obtained after crypto-compression. The works using the DCT compression ([17], [20], [21]), though offering good compression ratios, bring remarkable changes on the image. Compression based on FMT, AC,... solved this limit ([3], [4], [16]). However, when loss-less compression is needed, like in telemedicine, it is difficult to have good compression ratio. Moreover, encryption used in all these crypto-compression systems are either based on Number Theory or Chaos Theory. Chaos-based schemes are weakened by differential attacks for example; generally, they lack robustness and security [26]. Number Theory based schemes require keys of very large size, which is a problem in practice. Also, they will be easily solved by the quantum computer. So, new ways of securing images are explored and are still actual.

Elliptic Curve Cryptography (ECC), which has not yet been used for image crypto-compression, offers smaller key sizes: cryptosystems based on elliptic curves are secured with 160-bit keys when RSA uses 1024-bit keys for an equivalent level of security. This is an advantage for systems using smart cards where the memory space is very limited. In addition,

C.D.D. is a PhD Student at the University of Ngaoundere, P.O. box 455, Ngaoundere (corresponding author; phone: +237 676 09 39 77; Email: cidjeu@gmail.com)

D.T. is an Associate Professor at the Department of Mathematics and Computer Science of the University of Ngaoundere, P.O.box 455, Ngaoundere (Email: tieudjo@yahoo.com)

computing algorithms based on elliptic curves are faster. They offer more efficient implementations and have a larger rate of key generation and exchange [6]. So, using ECC in image crypto-compression can bring a remarkable improvement in terms of integrity, compression rate and execution time.

In this paper, we explore the use of elliptic curves in image processing. After compressing an image through JPEG, a Koblitz's type process [13] is described and adapted to transform images into sequences of points of an elliptic curve. This allows the manipulation of images as points on elliptic curves. In addition, we describe how to apply the Linear Feedback Shift Register (LFSR) on such points to produce an encrypted image. The decryption is also described. Finally, the algorithm is implemented on sample images and the quality of encrypted-decrypted image is analyzed. From the values of the entropy, image quality and PSNR, it's appears that the proposed scheme offers better integrity.

II. Preliminaries

A) Elliptic Curve Cryptography (ECC)

Let K be a field of characteristic different from 2 and 3. An elliptic curve E over K is the set of points

$$E = \{O\} \cup \{(x, y) \in K \times K, y^2 = x^3 + ax + b\}$$

where O is a specific point called the point at infinity. Fig. 1 below shows the elliptic curve $y^2 = x^3 - 2x - 2$ on the real field R.

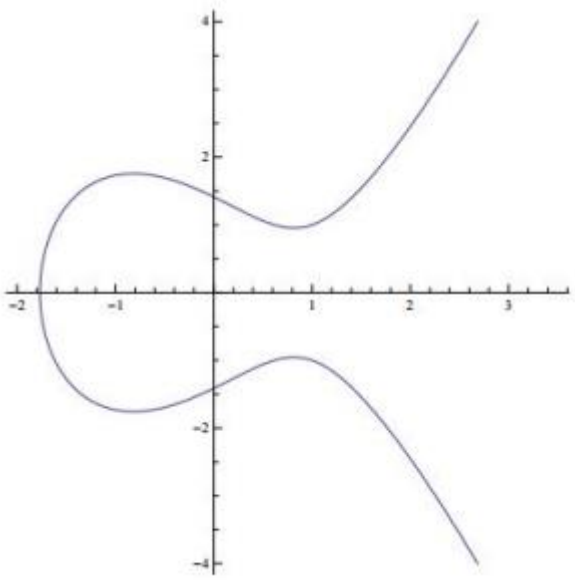


Fig. 1: Graph of the elliptic curve $y^2 = x^3 - 2x - 2$ on the real field R

Any point P can be represented by his projective coordinates which is a triple ($X_P : Y_P : Z_P$) corresponding to the affine coordinates (X_P/Z_P , Y_P/Z_P) if Z_P is non-zero, and O if Z_P is zero. Elliptic Curve Cryptography (ECC) is the use of elliptic curves for cryptographic purposes. Several cryptographic schemes have been built on elliptic curves. Among them, there are encryption schemes such as Elliptic Curve Integrated Encryption Scheme (ECIES) [6], [15]; key agreement protocols like Elliptic Curve Diffie-Hellman (ECDH) key exchange [6]; digital signatures protocols like Elliptic Curve Digital Signature Algorithm (ECDSA) and Edwards-curve Digital Signature Algorithm (EdDSA) [6]. Elliptic curves are also used for pseudo-random generators [6], integer factorization, etc. Nowadays, ECC is most used for key exchange.

B) Transforming a character to a point of an elliptic curve

For ECC to be applied directly on any data, these data as to be transformed to points on elliptic curve. In [13], Koblitz described a process to transform a character to a point of an elliptic curve. A character is seen as an integer m, such that $0 \leq m \leq M \in N$. For example, letters (A to Z) are numbers between 0 and 25. For a given character m, Algorithm 1 computes a pair (x, y) which is a point of an elliptic curve, representing the given character. Assume that we have a finite field Fq such that q is on the form $q = p^r$, p prime, $r > 0$; and $q \geq M\,k + 1$, where k is generally set to 30 or 50.

Given the curve $y^2 = x^3 + ax + b$ over the finite field Fq and given a character represented by an integer m.

Compute for each j = 1,...,k,

$$mk + j$$

Let x be the corresponding element of mk + j in Fq.

For such x, we compute $y^2 = f(x) = x^3 + ax + b$ and find a square-root for f(x). If there exists a y such that $y^2 = f(x)$, the point of the elliptic curve representing m is Pm = (x, y). If there is no square-root for f(x) for the current j, we jump to the next j. With k = 30 or k = 50 the algorithm always return a good result [13]. This process is detailed in Algorithm 1.

Algorithm 1 Transform a character to a point of an EC

Require: a character m, $\mathbb{F}_q$, k, a, b

Ensure: a pair $(x, y) \in \mathbb{F}_q \times \mathbb{F}_q$ representing m

1. j=1
2. while $j \leq k$
3. compute $\tilde{x} = mk + j$
4. write $\tilde{x}$ with r digits $m_{r-1}...m_1m_0$
5. compute $x = \sum_{i=0}^{r-1} m_i g^i \in \mathbb{F}_q$, where g is a generator of $\mathbb{F}_q$
6. compute $y^2 = f(x) = x^3 + ax + b$, and find a square-root for $f(x)$
7. if there exists a y such that $y^2 = f(x)$, then return $P_m = (x, y)$

else, increment j by 1.

From Algorithm 1, given a point (x, y) representing a character, this initial character m can be recovered by computing $\left\lfloor \frac{(\tilde{x}-1)}{k} \right\rfloor$, where $\lfloor v \rfloor$ represents the integer part of v and $\tilde{x}$ is the integer which corresponds to x in the equivalence between the integers and the elements of Fq.

C) Image crypto-compression

Image crypto-compression consists in joining encryption and compression to produce a secure and lighter data.

JPEG is the standard in image compression. The best performances are achieved with JPEG2000 which uses Discrete Wavelet Transform (DWT) instead of Discrete Cosine Transform (DCT). With JPEG2000, we can obtain loss-less compression and better compression rate and quality, compare to JPEG1991 (using DCT) [22]. The performance of an image compression algorithm can be measured by the compression ratio and the Peak Signal to Noise Ratio (PSNR). The compression ratio allows to see how far the image size has been reduced after compression; it can be computed by the following formula [5]:

$$t = 1 - \frac{Size_{final}}{Size_{initial}} \quad (1)$$

$Size_{initial}$ and $Size_{final}$ represent the size of the image respectively before and after compression. If the compression ratio is close to 1, it means the size has considerably reduced during processing; if on the other hand it is close to zero, the size has not greatly changed.

The Peak Signal to Noise Ratio (PSNR) evaluates the distortion brought on the image by the process. For a given M × N image, PSNR is defined in [4] as:

$$PNSR = 10Log_{10}(N_g^2/MSE) \quad (2)$$

where Ng is the maximal pixel value and *MSE* the Mean Square Error, given by:

$$MSE = \frac{\sum_{n=1}^{N}\sum_{m=1}^{M}(I_{original}(m,n) + I_{compressed}(m,n))^2}{M \times N} \quad (3)$$

The lower the PNSR is, the more the image has been distorted.

Concerning the encryption part, many ways of performing image encryption are possible. One of them is the stream encryption using Linear Feedback Shift Register (LFSR) [19]. Here, encryption is done pixel by pixel. For each pixel p(n) of the original image and for a key of length k, the value p(n)' of the corresponding pixel p(n) in the ciphered image is computed in the following way:

$$p(n)' = p(n) + \sum_{i=1}^{i=k} \alpha(i)p'(n-i)$$

with n ∈ [k, N], N being the number of pixels of the image, k ∈ [1, n] and α(i) a sequence of coefficients generated by the key. Figure 2 gives an overall view of the LFSR image encryption method.

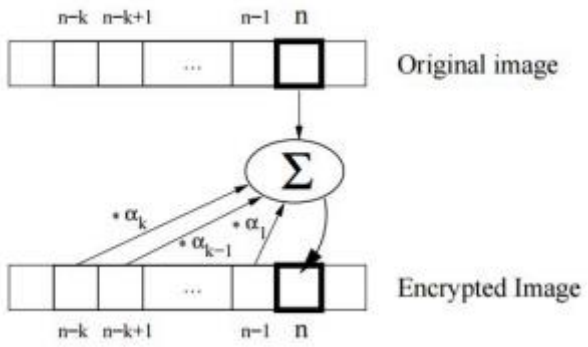


Fig. 2: Image encryption by LFSR [19]

Image encryption can be evaluated by the entropy, the histogram and the image quality after decryption [5]. The entropy indicates how far the information is random. It is calculated by the following formula [5]:

$$H(I) = -\sum_i P_i log_2(P_i) \quad (4)$$

where the Pi are the distribution of gray levels in the image. Two close entropies indicate two close information.

The histogram shows how far the information contained in the encrypted image is different from the one on the original image. They are many ways of showing the histogram of an image. We used the online tool http://www.dcode.fr/histo_ gramme-image. The number of modified pixels values during the encryption is another factor of the integrity of the process. We calculate the percentage of pixels values not modified to evaluate image quality after reconstitution.

III. Crypto-compression of image as points of an elliptic curve

A) Transforming image to points of an Elliptic curve

Algorithm 1 above describes how to transform a character into a point of an elliptic curve. An image is a sequence of integer values between 0 and 255.

Given an image, we present below how to generate the sequence of points of an elliptic curve representing the given image. Algorithm 2 shows how the integer values between 0 and 255 are represented on a given elliptic curve.

Algorithm 2 Transform pixel values to points on EC (PointsEC)

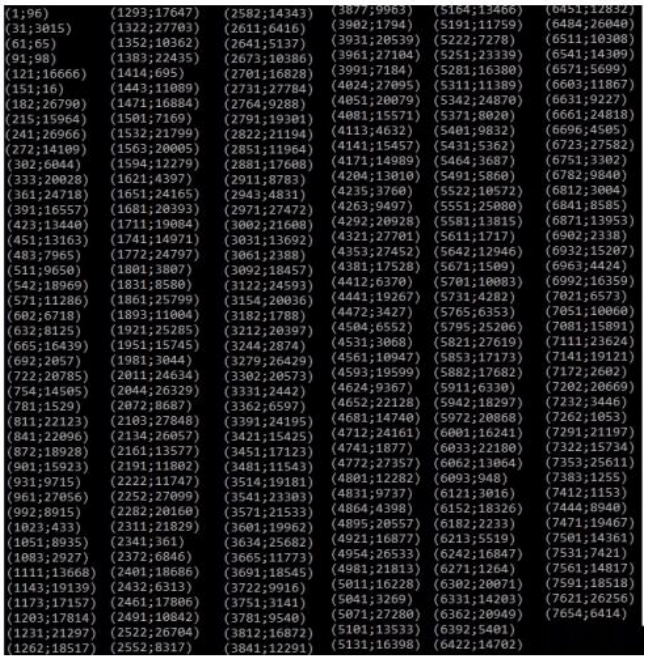
Require: an elliptic curve E over $\mathbb{F}_q$

Ensure: a sequence of points $(x, y) \in \mathbb{F}_q \times \mathbb{F}_q$ representing the 256 pixel values

1. points=[]
2. For each pixel value m between 0 and 255
   2.1 execute Algorithm 1 to find P_m
   2.2 add P_m to list
3. Return points

Fig. 3 presents a list of the 256 points obtained after

executing Algorithm 2. These points represent the pixels values from 0 to 255 on the elliptic curve of equation $y^2 = x^3 + 17x + 19$ over F_{174}.



(1;96)	(1293;17647)	(2582;14343)	(3877;9983)	(5164;13466)	(6451;12832)
(31;3015)	(1322;27703)	(2611;6416)	(3902;1794)	(5191;11759)	(6484;26040)
(61;65)	(1352;10362)	(2641;5137)	(3931;20539)	(5222;7278)	(6511;10308)
(91;98)	(1383;22435)	(2673;10386)	(3961;27104)	(5251;23339)	(6541;14309)
(121;16666)	(1414;695)	(2701;16828)	(3991;7184)	(5281;16380)	(6571;5699)
(151;16)	(1443;11089)	(2731;27784)	(4024;27095)	(5311;11389)	(6603;11867)
(182;26790)	(1471;16884)	(2764;9288)	(4051;20079)	(5342;24870)	(6631;9227)
(215;15964)	(1501;7169)	(2791;19301)	(4081;15571)	(5371;8020)	(6661;24818)
(241;26966)	(1532;21799)	(2822;21194)	(4113;4632)	(5401;9832)	(6696;4505)
(272;14109)	(1563;20005)	(2851;11964)	(4141;15457)	(5431;5362)	(6723;27582)
(302;6044)	(1594;12279)	(2881;17608)	(4171;14989)	(5464;3687)	(6751;3302)
(333;20028)	(1621;4397)	(2911;8783)	(4204;13010)	(5491;5860)	(6782;9840)
(361;24718)	(1651;24165)	(2943;4831)	(4235;3760)	(5522;10572)	(6812;3004)
(391;16557)	(1681;20393)	(2971;27472)	(4263;9497)	(5551;25080)	(6841;8585)
(423;13440)	(1711;19084)	(3002;21608)	(4292;20928)	(5581;13815)	(6871;13953)
(451;13163)	(1741;14971)	(3031;13692)	(4321;27701)	(5611;1717)	(6902;2338)
(483;7965)	(1772;24797)	(3061;2388)	(4353;27452)	(5642;12946)	(6932;15207)
(511;9650)	(1801;3807)	(3092;18457)	(4381;17528)	(5671;1509)	(6963;4424)
(542;18969)	(1831;8580)	(3122;24593)	(4412;6370)	(5701;10083)	(6992;16359)
(571;11286)	(1861;25799)	(3154;20036)	(4441;19267)	(5731;4282)	(7021;6573)
(602;6718)	(1893;11004)	(3182;1788)	(4472;3427)	(5765;6353)	(7051;10060)
(632;8125)	(1921;25285)	(3212;20397)	(4504;6552)	(5795;25206)	(7081;15891)
(665;16439)	(1951;15745)	(3244;2874)	(4531;3068)	(5821;27619)	(7111;23624)
(692;2057)	(1981;3044)	(3279;26429)	(4561;10947)	(5853;17173)	(7141;19121)
(722;20785)	(2011;24634)	(3302;20573)	(4593;19599)	(5882;17682)	(7172;2602)
(754;14505)	(2044;26329)	(3331;2442)	(4624;9367)	(5911;6330)	(7202;20669)
(781;1529)	(2072;8687)	(3362;6597)	(4652;22128)	(5942;18297)	(7232;3446)
(811;22123)	(2103;27848)	(3391;24195)	(4681;14740)	(5972;20868)	(7262;1053)
(841;22096)	(2134;26057)	(3421;15425)	(4712;24161)	(6001;16241)	(7291;21197)
(872;18928)	(2161;13577)	(3451;17123)	(4741;1877)	(6033;22180)	(7322;15734)
(901;15923)	(2191;11802)	(3481;11543)	(4772;27357)	(6062;13064)	(7353;25611)
(931;9715)	(2222;11747)	(3514;19181)	(4801;12282)	(6093;948)	(7383;1255)
(961;27056)	(2252;27099)	(3541;23303)	(4831;9737)	(6121;3016)	(7412;1153)
(992;8915)	(2282;20160)	(3571;21533)	(4864;4398)	(6152;18326)	(7444;8940)
(1023;433)	(2311;21829)	(3601;19962)	(4895;20557)	(6182;2233)	(7471;19467)
(1051;8935)	(2341;361)	(3634;25682)	(4921;16877)	(6213;5519)	(7501;14361)
(1083;2927)	(2372;6846)	(3665;11773)	(4954;26533)	(6242;16847)	(7531;7421)
(1111;13668)	(2401;18686)	(3691;18545)	(4981;21813)	(6271;1264)	(7561;14817)
(1143;19139)	(2432;6313)	(3722;9916)	(5011;16228)	(6302;20071)	(7591;18518)
(1173;17157)	(2461;17806)	(3751;3141)	(5041;3269)	(6331;14203)	(7621;26256)
(1203;17814)	(2491;10842)	(3781;9540)	(5071;27280)	(6362;20949)	(7654;6414)
(1231;21297)	(2522;26704)	(3812;16872)	(5101;13533)	(6392;5401)	
(1262;18517)	(2552;8317)	(3841;12291)	(5131;16398)	(6422;14702)	

Fig. 3: List of points corresponding to the 256 pixels values

Algorithm 3 shows, using algorithm 2, how to obtain a sequence of points of an elliptic curve, representing a given image.

Algorithm 3 Transform an image to points on EC

Require: an image I

Ensure: a sequence of points $(x, y) \in \mathbb{F}_q \times \mathbb{F}_q$ representing the given image

1. Define an elliptic curve E on the form $y^2 = x^3 + ax + b$ over $\mathbb{F}_q$
2. computes points=PointsEC(E)
3. imageEC=[]
4. For each pixel m in I
   4.1 add points[m] to imageEC
5. Return imageEC

By using Algorithm 3, an image can be viewed as points of an elliptic curve, fixed in advance. Consequently, what is done on images can be done on points of an elliptic curve and what is done on elliptic curves can be applied on images.

B) Compression and EC

Any image can be transformed to points on an EC. By compressing the image in advance, we obtain a lighter image. So, we will first perform a loss-less JPEG compression before transforming the compressed image to points on EC as presented on Fig. 4.

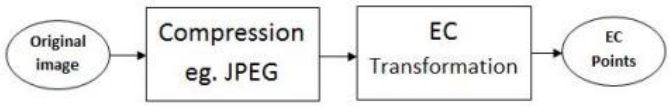


Fig. 4: Compressing and transforming an image to points of EC

C) Encryption/decryption of points of EC with LFSR

Let P be a sequence of N points of an elliptic curve, representing a given image. The encrypted point P(n)' corresponding to the n-th point P(n) using the Linear Feedback Shift Register (LFSR) can be performed as follows:

$$P'(n) = P(n) + \sum_{i=1}^{i=k} \alpha(i) * P'(n-i)$$

with n ∈ [0, N − 1] and α(i) a sequence of coefficients generated by the key. Note that the ∗ here is the multiplication of the points of an elliptic curve by a scalar. For decryption, the following formula is applied:

$$P(n) = P'(n) - \sum_{i=1}^{i=k} \alpha(i) * P'(n-i)$$

Fig. 5 presents the encryption/decryption of images as points on elliptic curve (EC points).

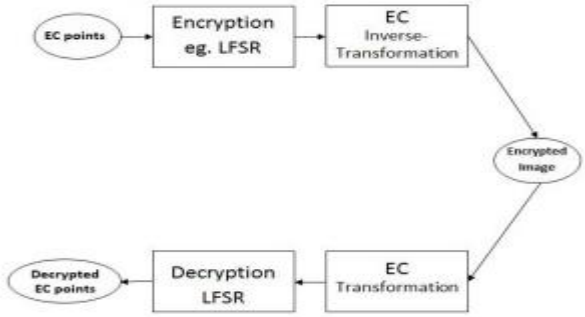


Fig. 5: Encryption/decryption process

Below (Fig. 6) is the whole process of image crypto-compression using JPEG, elliptic curves and LFSR.

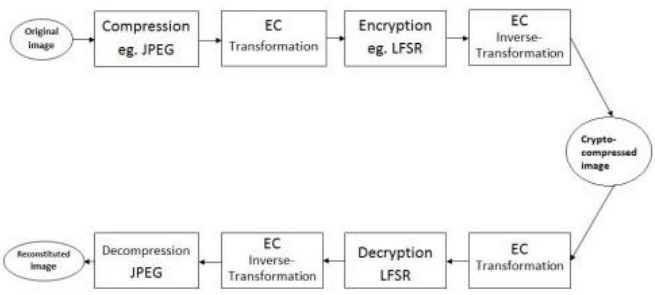


Fig. 6: crypto-compression with JPEG, EC and LFSR

IV. Implementation and Discussion

The implementation has been made under the computer algebra system SAGE-5.1. It contains many aspects of mathematics, including algebra, combinatorics, graph theory, numerical analysis, number theory, statistics and cryptography [23]. We used a core i5 (2.3hz) computer, with RAM 4Go. Transformation of images into points on elliptic curve, encryption and decryption are fully implemented. We made computations on two sample images, ”lena” and ”arch” [5].

A) Results and comparison

Fig. 7 presents two images compressed with loss-less JPEG, the encrypted and the decrypted images.

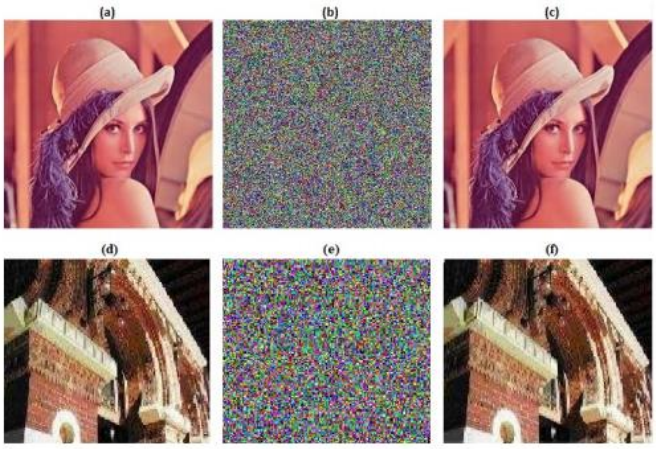


Fig. 7: (a),(d):compressed image ”lena”,”arch”;
(b),(e):encrypted image ”lena”,”arch”;
(c),(f):decrypted image ”lena”,”arch”

A sequence of points representing a portion of the compressed image “lena” is shown on Figure 8. On this figure, the points are represented by triples, which are their projective coordinates.

Fig. 8: Sequence of points related to the compressed image “lena”

Fig. 9 shows a portion of the sequence of points representing the encrypted image “lena”. The points on this sequence are different from those of the original image, meaning that the information has really been transformed.

Fig. 9: Sequence of points related to the encrypted image “lena”

Fig. 10 is a part of the decrypted image “lena”. We can see that this sequence of points is the same than the one of the original images. This means that ECC did not make any changes to the image.

Fig. 10: Sequence of points related to the decrypted image “lena”

Fig. 11 shows the histograms of the compressed, the ciphered and the deciphered images respectively of the image “lena”.

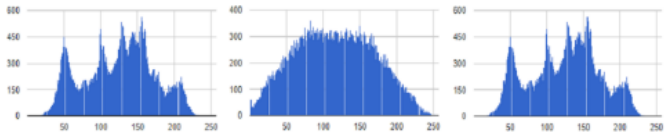


Fig. 11: Histograms of the image “lena” before crypto-compression, after crypto-compression and after reconstitution

The difference between the histograms of the original and the encrypted image indicates that the information has been really modified by the crypto-compression process. Two identical histograms for the original and the reconstituted image reveals that the information has not been destroyed.

On Fig. 12, histograms of the compressed, the ciphered and the deciphered images respectively of the image “arch” are presented.

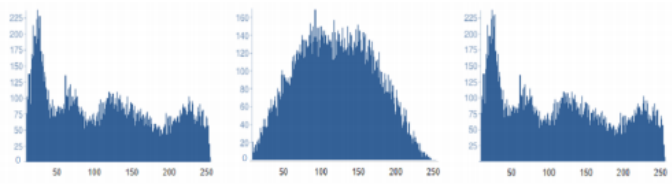


Fig. 12: Histograms of the image “arch” before crypto-compression, after crypto-compression and after reconstitution

Here also, the information contained in the image have been well transformed and restored by the crypto-compression and the inverse crypto-compression processes.

Table I below presents a comparison between the crypto-compression scheme implemented in this paper (JPEG and LFSR on elliptic curves), and two other systems (JPEG-AES and FMT-AES). We also compute the compression rate without JPEG (EC-LFSR), to appreciate the compression gain with JPEG.

Table I: Comparison with some encryption-description systems

”lena”	Entropy	Quality	PSNR	CPU Time	Compression rate
Original image	7,589	0.91	/	/	/
JPEG-AES	7,083	0.89	35.351	9.95	91%
FMT-AES	6.981	0.89	34.879	3.15	90%
JPEG-EC-LFSR	7,335	0.90	47.892	13.30	91%

As presented on Table I, PNSR and image quality are greater with the JPEG-EC-LFSR, which means that the distortion on the original image is less important. This can explain why the entropy is the nearest to the original one, compare to those obtain with the two other systems (JPEG-AES and FMT-AES).

In Table II, image quality and entropy of the crypto-compressed image is the same than in the original image.

Table II: Comparison with some encryption-description systems

”arch”	Entropy	Quality	PSNR	CPU Time	Compression rate
Original image	7,887	0.92	/	/	/
JPEG-AES	8,271	0.89	54.265	11.15	84%
FMT-AES	8.181	0.89	53.791	3.37	86%
JPEG-EC-LFSR	7,887	0.92	44.175	13.47	84%

All these results show a better integrity preservation. However, the CPU time is more important than those of the two previous schemes.

B) Conclusion and perspectives

Combining loss-less JPEG compression, Elliptic Curve Cryptography and LFSR encryption, we obtain a better level of security and integrity, compare to JPEG-AES and FMT-AES. However, the CPU time needed in encryption/decryption is more important. This weakness can be removed by using a lighter encryption scheme on elliptic curves. The gain in integrity and security despite the time factor is particularly interesting for some images. This can for example be very useful for certain medical images which contain very sensitive data and do not always need a real time processing.

In this paper, LFSR is used to encrypt and decrypt a standard image as points of an elliptic curve. Any other compression process or cryptosystem can be applied to images considered as points of an elliptic curve.

REFERENCES

[1] Abdmouleh M.K., Bouhlel M.S., E ective Crypto-compression Scheme for Medical Images, International Jour-nal of Signal Processing, (2017).

[2] Bahrami S. and Naderi M., Image Encryption Using a Lightweight Stream Encryption Algorithm, Advances in Multimedia (2012).

[3] Benabdellah M, Majid H.M., Zahid N., Regragui F. and Bouyakhf E.H., Encryption-Compression of still images using the FMT transformation and the DES algorithm, International Journal of Computer Sciences and Telecommunications, No. 4, (2006).

[4] Benabdellah M, Majid H.M., Zahid N., Regragui F. and Bouyakhf E.H., Encryption-compression of images based on FMT and AES algorithm, International Journal Applied Mathematical Sciences, Vol. 1, (2007).

[5] Bebabdelah M., Outils de compression et de crypto-compression: applications aux images xes et video, These de doctorat, Universit Mohammed V-Agdal, (2007).

[6] Bos J. W., Halderman J. A., Heninger N.,Moore J., Naehrig M.and Wustrow E., Elliptic Curve Cryptography in Practice, IACR, (2013).

[7] Dridi M., Bouallegue B., Hajjaji M. A., and Mtibaa A., An enhancement crypto-compression scheme for image Based on chaotic system, IJAER, Vol. 11, No. 7, pp 4718-4725, (2016).

[8] Dubal M. and Deshmukh A., On Pseudo-random Number Generation Using Elliptic Curve Cryptography, Security in Computing and Communications, Vol. 377, pp. 77-89, (2013).

[9] Fterich S. and Ben Amar C., Crypto-Compression d'Images Fixes Par la methode de Quadtree optimisee et AES, Compression et Representation des Signaux Audiovisuels, (2004).

[10] Gao H., Zhang Y., Liang S. and Li D., A new chaotic algorithm for image encryption, Chaos, Solitons and Fractals, Vol. 29, (2006).

[11] Jalel H., Mohamed A., Ben F., Mounir S. and Abdennaceur K., Crypto-compression of images based on chaos, IEEE, (2013).

[12] Klein A., Linear Feedback Shift Registers, Stream Cipher (chap 2), Springer, (2013).

[13] Koblitz N., A Course in Number Theory and Cryptography - 2nd ed., Springer-Verlag, (1994).

[14] Lafourcade P., Security and Cryptography just by images, Universit Joseph Fourier, Verimag DCS, (2009).

[15] Mart nez V.G., Encinas L.H. and Avila C.S., A Survey of the Elliptic Curve Integrated Encryption Scheme, Journal of Computer Science and Engeneering, Vol. 2, (2010).

[16] Masmoudi A., Puech W., Lossless chaos-based crypto-compression scheme for image protection, ITE Image Processing, Vol. 8, (2014).

[17] Meraoubi H., Brahimi Z., Ait Saadi K. et Zemouri A., Un systeme de crypto-compression des images medicales base sur la DCT 2 2-IDS et l'AES, 1st ISESC, (2005).

[18] Miao Z., Xiaojun T., Joint image encryption and compression scheme based on IWT and SPIHT, Optics and Lasers in Engineering, Vol. 90, pp. 254-274, (2017).

[19] Puech W. and Rodrigues J.M., A new crypto-watermarking method for medical images safe transfer, In EUSIPCO'04, Vienne, Autriche, (2004).

[20] Puech W. and Rodrigues J.M., Crypto-Compression of medical images by selective encryption of DCT, In EUSIPCO'05, Antalya, Turquie, Septembre (2005).

[21] Puech W., Rodrigues J.M. et Develay-Morice J.E., Transfert sécurisé d'images médicales par codage conjoint: cryptage sélectif par AES en mode par flot et compression JPEG, Traitement du signal (TS), numéro spécial Traitement du signal applique à la cancérologie, Vol. 23, No. 5, (2006).

[22] Salomon D., Data Compression: The Complete Reference - Fourth Edition, Springer, (2007).

[23] Stein W., Sage Tutorial - Release 8.1, The Sage Development Team, (2017).

[24] Waghmare A., Bhagat A., Surve A., Kalgutkar S., Chaos Based Image Encryption and Decryption, IJAR-CCE, Vol. 5, (2016).

[25] Xiaoyong J., Sen B., Guibin Z. and Bing Y., Image encryption and compression based on the generalized knights tour, discrete cosine transform and chaotic maps, Multimedia Tools and Applications, Vol. 76, Issue 10, pp. 1296512979, (2017).

[26] Xingyuan W., Xiaojuan W., Jianfeng Z. and Zhenfeng Zhang, Chaotic encryption algorithm based on alter-nant of stream cipher and block cipher, Nonlinear Dynamics, Vol. 63, Issue 4, pp 587597, (2011).