Security Enhancement in Elliptic Curve Encryption of Amazigh alphabet using Genetic Algorithm

Fatima Amounas

R.O.I Group, Computer Sciences Department, Moulay Ismail University, Faculty of Sciences and Technics, Errachidia, Morocco.

f_amounas@yahoo.fr

Abstract
With the rapid development of technology, Information Security is becoming much more important in data transmission. Cryptography is the practice and study of encrypting/decrypting information. Due to its importance, several cryptography techniques and algorithms are adopted by many authors to secure the data, but still there is a scope to improve the previous approaches. This paper introduce a security enhancement on the elliptic curve cryptosystem, it suggests the use of Genetic Algorithms in the encryption process to make the cryptosystem more secure. The proposed method is implemented in Java and applied for the encryption and decryption of a text file. As result, our proposed algorithm is fast and secure to be used for encryption of Amazigh Alphabet.

Keywords: Elliptic Curve, Genetic Algorithm, Secret key, Encryption, Decryption, Amazigh Alphabet.

I. Introduction
In the recent years, computer world security, integrity, confidentiality of the organization’s data is the most important issue. Cryptography is an art of writing and reading the secret information. It uses mathematics in science to protect the information. In order to protect valuable information from undesirable users or against illegal reproduction and modifications, various types of cryptographic schemes are needed.

There are two types of cryptographic schemes: symmetric cryptography and asymmetric cryptography. The symmetric scheme uses the same key for encryption and decryption. Symmetric key cryptography is used for private encryption of data to achieve high performance. For e.g AES, IDEA, DES, ... In asymmetrical cryptography, two keys are used, one for encryption, known as the public key and the other for decryption, known as the private key. The most of standards that use public key technique for encryption and decryption are based on RSA cryptosystem and ECC. The main attraction of ECC is that it can provide better performance and security for small key size, in comparison of RSA cryptosystem (Darrel Hankerson et al. 2004).

Recently in Morocco, security of Amazigh characters has become a growing field in which several researchers work. In order to protect Amazigh information, several research efforts have been made and some research works of Tifinagh characters are published (F.Amounas et al. 2012, A. El Ghazi 2014, F.Amounas 2015). In this context, we proposed a novel approach of encrypting Amazigh characters based on elliptic curve using Genetic Algorithm. The proposed approach uses the concept of Genetic Algorithms to enhance security of the cryptosystem. For more detail on Amazigh language, we refer to (L. Zenkouar 2004, Y. Es saady et al. 2011).
The remainder of this paper is arranged as follows: Section 2 gives some background information of elliptic curve cryptography and Genetic Algorithms. Section 3 discusses related works. Section 4 focuses on the proposed method of ECC encryption using Genetic Algorithm. Section 5 covers implementation of the algorithm in Java. The last section describes the concluded part of the work.

II. Related Works
In the literature, only few genetic algorithm based encryption have been proposed. Authors in (A.Tragha et al. 2006) describe a new symmetrical block ciphering system, which generates a session key in a random process. The block sizes and the key length are variable and can be fixed by the user at the beginning of the ciphering. In (Ankita Agarwal 2012), Ankita proposed a method based on Genetic Algorithm (GA) which is used to produce a new encryption method by exploitation the powerful features of the Crossover and Mutation operations of (GA). Recently, the author in (Sonia Goyat 2012) proposes the use of GA in Cryptography. The work explores the different techniques of cryptography in order to prove that the natural selection based techniques are as good as the rigorous mathematical techniques. After two years, the authors in (Sindhuja K and al. 2014) propose a new method to encrypt data using right shift, matrix addition, modulo operation and genetic operations. The results show that the presented method is satisfied the goals that are need in any encryption method. Furthermore, Sania Jawaid and Adeeba Jamal propose a method of key generation based on the theory of natural selection (Sania Jawaid et al. 2014). The current paper is an attempt to present a new approach for encrypting and decrypting Amazigh characters based on elliptic curve cryptography in such a way that the new approach can make use of Genetic Algorithm to achieve higher level of security.

III. Background detail
In this section, we give a brief overview on elliptic curve cryptography and Genetic Algorithms.

A. Elliptic Curve Cryptography
Elliptic Curve Cryptography (ECC) was introduced in 1986 by Victor Miller and Neil Koblitz as an alternative to other public key cryptosystem present such as RSA (R.Rivest and al. 1978) and ElGamal (T.Elgamal 1985) Cryptosystems. The mathematical background of ECC is more complex and thus it provides greater security and more efficient performance than other public key cryptosystems (Lawrence C.Washington 2008).

Let \( F_p \) be a finite field where \( p \) is prime. Let \( E \) be an elliptic curve over \( F_p \). The curve \( E \) takes the general form as:

\[
y = x^3 + ax + b \mod p
\]  

Where \( x, y \) are elements of the finite field \( F_p \) and \( a, b \) are in integers satisfying:

\[
4a^3 + 27b^2 \neq 0 \mod p
\]  

The elliptic curve can be used to construct an abelian group \( E(F_p) \) with identity element \( \Omega \) called the point infinity. The standard representation of points on an elliptic curve \( E \) over prime fields is the affine coordinates (N. Koblitz and al. 2000). A point \( P \in E(F_p) \) is represented as \( P=(x,y) \) where it’s inverse is \( -P=(x,-y) \). Let \( P(x_1,y_1) \) and \( Q(x_2,y_2) \) are two points on EC. The point addition \( P+Q \) is computed using Eq. 3 and Eq. 4. The point doubling \( 2P \) is computed using Eq.5 and Eq.6.

\[
x_3 = t^2 - x_1 - x_2
\]
\[ y_3 = t(x_1 - x_j) - y_1 \] (4)

where

\[ t = \frac{y_2 - y_1}{x_2 - x_1} \]

\[ x_3 = t^2 - 2x_1 \] (5)

\[ y_3 = t(x_1 - x_j) - y_1 \] (6)

where

\[ t = \frac{3x_i^2 + a}{2y_i} \]

**B. Genetic Algorithms**

Genetic Algorithms (GAs) are a randomized search algorithms and optimization techniques guided by the principle of natural selection systems. The process of Genetic Algorithms (GA), usually, starts with a population which is randomly generated and is composed of several chromosomes (A Kumar and al. 2004). The chromosomes are either binary or hex number, depending on the type of population. GA’s transform the population of chromosomes, generated into a new generation. Various operators are applied when the population is generated for selection of individuals (A. Tragha and al. 2007). The individuals are selected based on their probability, various genetic operations and their fitness value. The set of operators usually consists of mutation, crossover and selection (S. N. Sivanandan and al. 2008).

- **Crossover:** Crossover is a genetic operator that helps in joining two chromosomes to form a new chromosome. The newly generated chromosome is called child which takes one part of chromosome from each parent. Crossover can be classified into following types:
  - Single point crossover: in this type of crossover, only one crossover point is chosen to generate new child.
  - Two point crossover: this type of crossover involves selecting two crossover points to generate new child.
  - Uniform crossover: in this type of crossover bits of child are uniformly taken from both the parents.

- **Mutation:** Mutation is a genetic operator which changes one or more bit values in a chromosome. It is performed on a child after crossover which guarantees the entire state-space will be searched. Mutation can be classified into following types:
  - Flipping of Bits: It involves selecting one or more bits of chromosome and inverting it.
  - Boundary Mutation: It involves randomly replacing chromosome with either lower or upper bound.
  - Non-Uniform Mutation: It is used to increase the probability that amount of mutation will go to 0 with the next generation.
  - Uniform Mutation: A chosen chromosome cell is replaced with a uniform random value whose range is selected by user.
  - Gaussian Mutation: It involves adding a unit gaussian random value to a chromosome cell.

- **Selection:** Selection is the stage of a GAs in which individual chromosomes are chosen from a population for recombination (or crossover). The chromosome with higher fitness value will be considered better. Selection can be classified into following types:
Roulette-wheel Selection: if procedure is repeated until there are enough selected individuals.

Stochastic Universal Sampling: if instead of a single pointer spun multiple times, there are multiple, equally spaced pointers on a wheel that is spun once.

Tournament Selection: it refers to repeatedly selecting the best individual of a randomly chosen subset.

Truncation Selection: it involves taking the best half, third or another proportion of the individuals.

For more details on the theory of Genetic Algorithms, we refer interested reader to (Bethany Delman 2004, Scott M. Thede 2004).

IV. Main Results
Our contribution in this paper is to introduce a new concept of ECC encryption to enhance the security of Amazigh alphabet by using Genetic Algorithms. In the proposed method Genetic Algorithm (GA) will be used in key generation process and in all modules. The main idea behind GAs is to make the key very complex. This approach combines the features of Genetic Algorithm and elliptic curve cryptography.

A. Algorithm (1) Generation of secure key
The process of generating the key with the help of genetic functions “Crossover” and “Mutation” has the following steps:

Step 1. Choose a random number k and compute kP_B.

Step 2. Transform the key into code point and imbed the character into point on elliptic curve. Convert the block in binary form.

Step 3. Perform crossover operation on the results blocks, to achieve good randomness among the key.

Step 4. After crossover operation the bits of the block are swapped again to permute the bit values using mutation operation.

Step 5. Repeat the steps 3-4 for m times.

Step 6. The final key is generated.
This key will be then used for encryption process.

B. Encryption
The encryption process involves the following steps:

Step 1. Read source file (Amazigh.txt) character by character. Then, imbed all characters into points on elliptic curve.

Step 2. Divide the plain text in 64 blocks with elements on EC.

Step 3. Arrange these blocks into 8 x 8 table format.

<table>
<thead>
<tr>
<th>P_{11}</th>
<th>P_{12}</th>
<th>P_{13}</th>
<th>P_{14}</th>
<th>P_{15}</th>
<th>P_{16}</th>
<th>P_{17}</th>
<th>P_{18}</th>
</tr>
</thead>
<tbody>
<tr>
<td>P_{21}</td>
<td>P_{22}</td>
<td>P_{23}</td>
<td>P_{24}</td>
<td>P_{25}</td>
<td>P_{26}</td>
<td>P_{27}</td>
<td>P_{28}</td>
</tr>
<tr>
<td>P_{31}</td>
<td>P_{32}</td>
<td>P_{33}</td>
<td>P_{34}</td>
<td>P_{35}</td>
<td>P_{36}</td>
<td>P_{37}</td>
<td>P_{38}</td>
</tr>
<tr>
<td>P_{41}</td>
<td>P_{42}</td>
<td>P_{43}</td>
<td>P_{44}</td>
<td>P_{45}</td>
<td>P_{46}</td>
<td>P_{47}</td>
<td>P_{48}</td>
</tr>
<tr>
<td>P_{51}</td>
<td>P_{52}</td>
<td>P_{53}</td>
<td>P_{54}</td>
<td>P_{55}</td>
<td>P_{56}</td>
<td>P_{57}</td>
<td>P_{58}</td>
</tr>
<tr>
<td>P_{61}</td>
<td>P_{62}</td>
<td>P_{63}</td>
<td>P_{64}</td>
<td>P_{65}</td>
<td>P_{66}</td>
<td>P_{67}</td>
<td>P_{68}</td>
</tr>
<tr>
<td>P_{71}</td>
<td>P_{72}</td>
<td>P_{73}</td>
<td>P_{74}</td>
<td>P_{75}</td>
<td>P_{76}</td>
<td>P_{77}</td>
<td>P_{78}</td>
</tr>
<tr>
<td>P_{81}</td>
<td>P_{82}</td>
<td>P_{83}</td>
<td>P_{84}</td>
<td>P_{85}</td>
<td>P_{86}</td>
<td>P_{87}</td>
<td>P_{88}</td>
</tr>
</tbody>
</table>

Step 4. Compute secure key K and encrypt the sequence of points using ECC encryption formula to obtain the encrypted points Q_i.
Step 5. Now divide into four sub matrices (M₁, M₂, M₃ and M₄). Then, convert each point into hex code. The result code is denoted Aᵢ.

M₁  M₂  M₃  M₄

<table>
<thead>
<tr>
<th>A₁₁</th>
<th>A₁₂</th>
<th>A₁₃</th>
<th>A₁₄</th>
<th>A₂₁</th>
<th>A₂₂</th>
<th>A₂₃</th>
<th>A₂₄</th>
<th>A₃₁</th>
<th>A₃₂</th>
<th>A₃₃</th>
<th>A₃₄</th>
<th>A₄₁</th>
<th>A₄₂</th>
<th>A₄₃</th>
<th>A₄₄</th>
</tr>
</thead>
<tbody>
<tr>
<td>A₅₁</td>
<td>A₅₂</td>
<td>A₅₃</td>
<td>A₅₄</td>
<td>A₆₁</td>
<td>A₆₂</td>
<td>A₆₃</td>
<td>A₆₄</td>
<td>A₇₁</td>
<td>A₇₂</td>
<td>A₇₃</td>
<td>A₇₄</td>
<td>A₈₁</td>
<td>A₈₂</td>
<td>A₈₃</td>
<td>A₈₄</td>
</tr>
</tbody>
</table>

Step 6. Generate the secure key K₁ using genetic operations like crossover and mutation (Algorithm 1).

Step 7. Let b= (bᵢ), where j is bit position (LSB→MSB), which decides which transformation, has to be performed on rows and columns of matrices.

Step 8. The value of b is verified: If b=1, row transformation is applied on the sub-matrices M₁ and M₄. If b=0, column transformation is to applied the sub-matrices M₂ and M₄.

The transformation represents genetic algorithm operations like selection, crossover and mutation.


Selection:
- Row transformation
  Consider 2¹ and 4ᵗʰ row as 1ˢᵗ pair of parent and other as 2ⁿᵈ pair of parent.
  Before selection:

<table>
<thead>
<tr>
<th>R₁₁</th>
<th>R₁₂</th>
<th>R₁₃</th>
<th>R₁₄</th>
</tr>
</thead>
<tbody>
<tr>
<td>R₂₁</td>
<td>R₂₂</td>
<td>R₂₃</td>
<td>R₂₄</td>
</tr>
<tr>
<td>R₃₁</td>
<td>R₃₂</td>
<td>R₃₃</td>
<td>R₃₄</td>
</tr>
<tr>
<td>R₄₁</td>
<td>R₄₂</td>
<td>R₄₃</td>
<td>R₄₄</td>
</tr>
</tbody>
</table>

After selection:

<table>
<thead>
<tr>
<th>R₂₁</th>
<th>R₂₂</th>
<th>R₂₃</th>
<th>R₂₄</th>
</tr>
</thead>
<tbody>
<tr>
<td>R₃₁</td>
<td>R₃₂</td>
<td>R₃₃</td>
<td>R₃₄</td>
</tr>
<tr>
<td>R₄₁</td>
<td>R₄₂</td>
<td>R₄₃</td>
<td>R₄₄</td>
</tr>
</tbody>
</table>

- Column transformation
  Consider 2ⁿᵈ and 4ᵗʰ column as 1ˢᵗ pair of parent and other as 2ⁿᵈ pair of parent.
  Before selection:

<table>
<thead>
<tr>
<th>C₁₁</th>
<th>C₁₂</th>
<th>C₁₃</th>
<th>C₁₄</th>
</tr>
</thead>
<tbody>
<tr>
<td>C₂₁</td>
<td>C₂₂</td>
<td>C₂₃</td>
<td>C₂₄</td>
</tr>
<tr>
<td>C₃₁</td>
<td>C₃₂</td>
<td>C₃₃</td>
<td>C₃₄</td>
</tr>
<tr>
<td>C₄₁</td>
<td>C₄₂</td>
<td>C₄₃</td>
<td>C₄₄</td>
</tr>
</tbody>
</table>
After selection:

\[
\begin{array}{cccc}
C_{21} & C_{22} & C_{23} & C_{24} \\
C_{31} & C_{32} & C_{33} & C_{34} \\
C_{41} & C_{42} & C_{43} & C_{44}
\end{array}
\]

**Crossover** (Two point crossover):

Now, we convert each element into binary form to get new data matrix B of \(4 \times m\). Then, we will randomly choose any two numbers from 1 to \(m\) as cross over point and implement crossover to data matrices. In our case, the crossover points are \((2, 8)\).

Crossover process

\[
\begin{array}{cccccccccccc}
b_{11} & b_{12} & b_{13} & b_{14} & b_{15} & b_{16} & \ldots & b_{17} & b_{18} & \ldots & b_{1m} \\
b_{21} & b_{22} & b_{23} & b_{24} & b_{25} & b_{26} & \ldots & b_{27} & b_{28} & \ldots & b_{2m} \\
b_{31} & b_{32} & b_{33} & b_{34} & b_{35} & b_{36} & \ldots & b_{37} & b_{38} & \ldots & b_{3m} \\
b_{41} & b_{42} & b_{43} & b_{44} & b_{45} & b_{46} & \ldots & b_{47} & b_{48} & \ldots & b_{4m}
\end{array}
\]

Result of crossover

\[
\begin{array}{cccccccccccc}
b_{11} & b_{12} & b_{13} & b_{14} & b_{15} & b_{16} & \ldots & b_{17} & b_{18} & \ldots & b_{1m} \\
b_{21} & b_{22} & b_{23} & b_{24} & b_{25} & b_{26} & \ldots & b_{27} & b_{28} & \ldots & b_{2m} \\
b_{31} & b_{32} & b_{33} & b_{34} & b_{35} & b_{36} & \ldots & b_{37} & b_{38} & \ldots & b_{3m} \\
b_{41} & b_{42} & b_{43} & b_{44} & b_{45} & b_{46} & \ldots & b_{47} & b_{48} & \ldots & b_{4m}
\end{array}
\]

**Mutation**

We will randomly choose any one numbers from 1 to \(m\) as mutation point and perform the mutation operation. Here the point is 5.

\[
\begin{array}{cccccccccccc}
b_{11} & b_{12} & b_{13} & b_{14} & b_{15} & b_{16} & \ldots & b_{17} & b_{18} & \ldots & b_{1m} \\
b_{21} & b_{22} & b_{23} & b_{24} & b_{25} & b_{26} & \ldots & b_{27} & b_{28} & \ldots & b_{2m} \\
b_{31} & b_{32} & b_{33} & b_{34} & b_{35} & b_{36} & \ldots & b_{37} & b_{38} & \ldots & b_{3m} \\
b_{41} & b_{42} & b_{43} & b_{44} & b_{45} & b_{46} & \ldots & b_{47} & b_{48} & \ldots & b_{4m}
\end{array}
\]

**Step 10.** Repeat step 9 for other matrices. The result matrices are denoted \(D_1, D_2, D_3\) and \(D_4\).

**Step 11.** Recombine the result matrices \(D_1, D_2, D_3\) and \(D_4\). Then, Convert each block to decimal form.

**Step 12.** Convert the result code into corresponding characters and store them to Enc.txt.

**C. Decryption**

The decryption process involves the following steps:

**Step 1.** Read the encrypted file character by character.

**Step 2.** Extract the first point \(P_1\) and applies his secret key \(n_B\) to compute the secure key \(K = n_B P_1\). Then, apply Genetic algorithms to generate secure key.

**Step 3.** Arrange the remaining blocks into \(8 \times 8\) table format.

**Step 4.** Divide this table into four sub matrices, noted \(D_1, D_2, D_3\) and \(D_4\). Then, convert each element into binary form.
Step 5. Let \( b = (b_j) \), where \( j \) is bit position (LSB→MSB), which decides which transformation has to be performed on rows and columns of matrices. Depending on the bit selected, the transformation is performed.

Step 6. Reverse the operations done in the encryption process (mutation, crossover, selection) get back data matrix \( M_i \).

Step 7. Convert each block into point on elliptic curve. Then, decrypt the sequence of points using ECC technique.

Step 8. Convert all points into characters and store them to Dec.txt.

V. Implementation and Result

The proposed method developed above is implemented in Java. In our case, the text file is taken as the input to the algorithm as shown in Figure 1. A JAVA Swing application is developed using Netbeans 7.1 to implement this methodology.

In this work, the text file is a collection of Amazigh sentences from the book "Le Petit Prince". This book has been translated in Amazigh language by Lahbib Fouad, a researcher at the Royal Institute of Amazigh Culture (IRCAM).

Files used in the program are shown in Table 1.

<table>
<thead>
<tr>
<th>Table 1. Files used in the program</th>
</tr>
</thead>
<tbody>
<tr>
<td>Filename</td>
</tr>
<tr>
<td>Amazigh.txt</td>
</tr>
<tr>
<td>Enc.txt</td>
</tr>
<tr>
<td>Dec.txt</td>
</tr>
</tbody>
</table>

Figure 1 shows the original input file to be encrypted.

![Figure 1. Text File as input of Algorithm “Amazigh.txt”](image)

The proposed system implementation is represented as a series of interface. The main interface of the application is illustrated in Figure 2. The user should enter the EC parameters (a, b, p), selects the base point on EC and chooses a random number. By clicking on “Generate Secure Key” button, the key generation process is performed using Genetic Algorithms.
To encrypt a text file “Amazigh.txt”, it is loaded in the application using the button “Select File”. The plaintext is encrypted with the help of the secure key generated as shown in Figure 3. A new encrypted-file is generated as shown in Figure 4.

At the receiver side, the decryption process can be performed by using the reverse orders of the encryption process to obtain the original file.
VI. Conclusion
This paper proposes a new approach for data security. It is based on ECC and uses the concept of genetic algorithms to achieve better of security. The important feature of the proposed method is that it is almost impossible to break the encryption algorithm without knowing the secure key value. Key generation algorithm provides a high quality security of data. The algorithm is being implemented on JAVA programming language and it works out perfectly. It is clear that this encryption method is satisfied the goals that are required in any encryption method for encrypt text files with the help of genetic operations. It is thus concluded that the proposed cryptosystem guarantee the confidentiality of Amazigh alphabet and provide better performance in this regard. Many improvements can be done in the future related to decrease matrices required to encrypt and decrypt Amazigh alphabet, and to decrease the cipher text size. On the other hand, compression technique may be used to reduce encryption time and optimize the overall algorithm performance.

References


