Elliptic Curve Cryptography

Gaganjeet Reen
ICM2015003

***Abstract*— In this paper, we take a secure Elliptic Curve E_p (a, b), where p is a large (greater than 5000) prime number. We then implement point addition and multiplication algorithms on this curve. Further, we pick a randomly chosen point P from the curve and calculate the order of the point. The point P should be such that Ord(P) = q, where q is a large prime. Then a large set of Q values are generated using randomly chosen k-bit(k = $\lceil \log_2 q \rceil$) strings.The decimal values of these chosen strings are all lesser than q-1. The following tests are performed on the Q values :-**
a) Whether the Q values are uniform-randomly distributed on the cyclic group generated by P.
b) Whether the Q values are uncorrelated even when there are high statistical correlations among the input k-bit strings.

I. INTRODUCTION

Elliptic-curve cryptography (ECC) is an approach to public-key cryptography based on the algebraic structure of elliptic curves over finite fields . Elliptic-curve cryptography requires smaller keys compared to non-Elliptic Curve cryptography to provide equivalent security.

The **elliptic curve** over Z_p , p > 3, is the set of all pairs (x, y) $\in Z_p$ which fulfill $y^2 \equiv x^3 + a*x + b$ mod p together with an imaginary point of infinity O, where a, b $\in Z_p$ and the condition $4*a^3 + 27*b^2 \neq 0$ modp. The definition of elliptic curve requires that the curve is **nonsingular**. Geometrically speaking, this means that the plot has no self-intersections or vertices, which is achieved if the discriminant of the curve $16(4a^3 + 27b^2)$ is nonzero.
However, simply defining an elliptic curve over a finite field is not enough. We need to find a curve which has a large cyclic group, which is needed for constructing a Discrete Logarithm Problem. We have identified the set of points needed for a group, however, we still need to define the group operation. The group operations defined on the obtained set of points is known as **Point Addition** and is denoted by the "+" symbol.

Point Addition :- Given two points and their coordinates, say P = (x1 , y1) and Q = (x2 , y2), we have to compute the coordinates of a third point R such that: P + Q == R i.e. (x1 , y1) + (x2, y2) == (x3 , y3)
Analytically :-

$$x3 = s^2 - x1 - x2\, modp \tag{1}$$

$$y3 = s(x1 - x3) - y1\, modp \tag{2}$$

where

$$s = (y2 - y1/x2 - x1)\, modp \tag{3}$$

if P $\neq$ Q (point addition) and

$$s = (3x1^2 + a/2y1)\, modp \tag{4}$$

if P = Q (**point doubling/point multiplication**)

Derivations of Point Addition Formulae :- Let P(x1,y1),Q(x2,y2) and S(x4,y4) be three points on a straight line intersecting the curve at points P , Q and S. The equation of the line through P and Q will be :-

$$y = y1 + s(x - x1) \tag{5}$$

Substituting this into

$$y2 = x3 + a * x + b \tag{6}$$

we get,

$$x^3 - s^2 * x^2 + (a + 2 * s^2 * x1 - 2 * s * y1)x + b - (s * x1 - y1)2 = 0 \tag{7}$$

The three solutions to that cubic equation give the x-coordinates x1,x2,x4 of the three points of intersection of the line with the curve.From Vieta's first formula, we see that the sum of those x-coordinates is s^2 so that

$$x1 + x2 + x4 = s^2. \tag{8}$$

When we reflect S over the x-axis, the x-coordinate does not change, so x3=x4. Thus,

$$x3 = s^2 - x1 - x2. \tag{9}$$

Using the equation of the line,

$$y4 = y1 + s * (x4 - x1) \tag{10}$$

Substituting x4 with x3 we get

$$y4 = y1 + s * (x3 - x1) \tag{11}$$

When we reflect S over the x-axis, the sign of the y-coordinate changes, i.e., y3=-y4. Thus,

$$y3 = s(x1 - x3) - y1 \tag{12}$$

.

Hasse's Theorem :- Given an elliptic curve E modulo p, the number of points on the curve is denoted by #E and is bounded by: p + 1 - $2\sqrt{p} \leq$ #E $\leq$ p + 1 + $2\sqrt{p}$.

Elliptic Curved Discrete Logarithm Problem(ECDLP) :- Given is an elliptic curve E. We consider a primitive element P and another element T . The Disrete Logarithm problem is finding the integer d, where 1 $\leq$ d $\leq$ #E, such that: P + P + + (d times)P = d P = T.

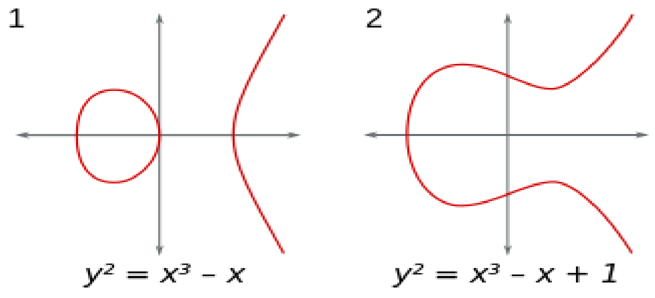

Fig. 1. Non singular elliptic curve

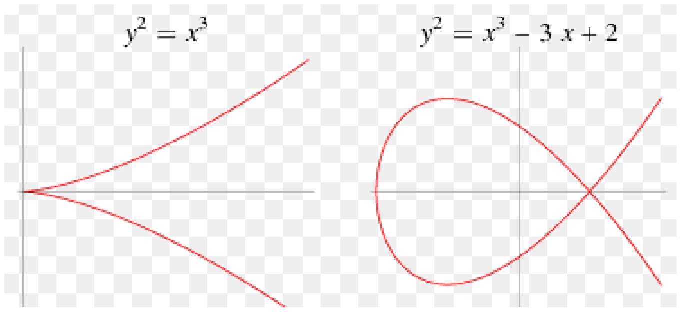

Fig. 2. Singular elliptic curve

II. Choosing a secure Elliptic Curve

Mathematically speaking, there is no proof that elliptic curves are actually "secure". But the same does apply to all other cryptographic algorithms, so we have to make do with the next best thing: since we cannot prove that any curve is "secure", we'll use curves that we do not know how to break. An important consequence is that only fully specified, standardised curves that have been analysed for years can claim to be "secure" in that sense. So, in practice, use one of the standard curves.
That being said, we can list a few basic criteria that "secure" curves must fulfill. Matching all these criteria does not guarantee security in any way, but if any of them is not matched, then the curve is definitely not secure.

A) The curve order (number of points on the curve) must be a large enough prime integer, or a multiple of a large enough prime integer. "Large enough" means here that the Elliptic Curve Discrete Logarithm problem should not be easily solvable by brute force i.e. a small sub-group attack should not be possible.This is required because according to Cauchys theorem, if G is a finite group and q is a prime number dividing the order of G then G contains an element of order q, and the order- q element generates an order q subgroup..
B) The curve should not be singular. This basically means that there should be no point of intersection on the elliptic curve.

III. Algorithms

A) Square and Multiply Algorithm

Input:
Base element x exponent H = $\sum_{i=0}^{t} h_i 2^i$ with hi $\in$ 0, 1 and h_t = 1 and modulus n
Output:
x^H mod n
Initialization:
r = 1

Algorithm 1 Square and Multiply

```
for  i = 0 DOWNTO t-1  do

    if h_i==1 then
        r = r*x modn
    end if
    x = x*x modn
end for
return r
```

B) Point Addition Algorithm

Input:
Points P,Q and p(large prime number for modulus)
Output:
resultant point on elliptic curve

Algorithm 2 Point Addition and Multiplication

```
Check validity of points P and Q
if P==Origin then
    return Q
else if Q==Origin then
    return P
else if P==Inverse of Q then
    return Origin
else if Q==Inverse of P then
    return Origin
else
    if P==Q then
        s = ( $3x1^2$ + a / 2y1 ) mod p
    else
        s = ( y2 - y1 / x2 - x1 ) mod p
    end if
    x = $s^2$ - P.x - Q.x mod p
    y = s(P.x - x ) - Q.y mod p
    return Point(x,y)
end if
```

C) Tonelli Shanks Algorithm

The TonelliShanks algorithm is used in modular arithmetic to solve for r in a congruence of the form $r^2 \equiv n \pmod{p}$, where p is a prime i.e. to find a square root of n modulo p.
The Algorithm :-
Inputs:

a) p, a prime
b) n, an element for which we need to check whether or not it is a quadratic residue under modulo p.

Outputs:

a) r in Z_p such that $r^2 \equiv n$ mod(p) (if it exists)

Algorithm 3 Tonelli Shanks

```
1: By factoring out powers of 2, find Q and S such that p - 1 =
   $Q2^S$ with Q odd
2: Search for a z in $Z_p$ which is a quadratic non-residue(found
   using Eulers Criterion)
3: $M \leftarrow S$
4: $c \leftarrow z^Q$
5: $t \leftarrow n^Q$
6: $R \leftarrow n^{(Q+1)/2}$
7: Loop :
   • If t = 0, return r = 0
   • If t = 1, return r = R
   • Otherwise, use repeated squaring to find the least i, 0 < i <
     M, such that $t^{2^i}$ = 1
   • Let b ← $c^{2^{M-i-1}}$ and set
     – M ← i
     – c ← $b^2$
     – t ← $tb^2$
     – R ← Rb
```

D) Finding order of a given point

Input:

- The point P
- a,b,p where p is a large prime and a and b are parameters of the ellptic curve such that a,b $\in Z_p$

Output:
The order of the point

Algorithm 4 Find Order

```
1: order=0
2: temp = Point
3: while Point P $\neq$ Inverse of Point P do
4:     order=order+1
5:     temp = Point_Addition(temp,P)
6: end while
7: return order+2
```

E) **Inverse using Fermats Theorem**
Input:
a,p where we need to computer inverse of a under modulo p and p is a prime number(more generally, a and p should be co-prime)
Output:
a^{-1}modp

Algorithm 5 Inverse using Fermats Little Theorem

```
1: return $a^{p-2}$modp
```

F) **Binary to Decimal Conversion**
Input:
k bit binary string
Output:
Decimal value of the binary string

Algorithm 6 Binary to Decimal

```
1: decimal = 0
2: i = 0
3: while binary $\neq$ 0 do
4:     dec = binary % 10
5:     decimal = decimal + dec*$2^i$
6:     binary = binary/10
7:     i = i+1
8: end while
9: return decimal
```

IV. IMPLEMENTATION

The code for the implementation of elliptic curve cryptography has been implemented using python programming language. As stated in the assignment, the word large has been defined to mean any value greater than 5000. The value selected for p is 15733 which according to our definition of large,is a large prime number.We choose the parameters a and b such that they ensure that the elliptic curve is non-singular i.e. $4*a^3 + 27*b^2 \neq 0$ modp. The values of a and b chosen are 1 and 3 respectively. The python modules which have been used are :-

- **secrets :-**This particular module is used to generate cryptographically secure random numbers.The secrets module is based on os.urandom() and random.SystemRandom(), which are the interface to the operating systems best source of cryptographic randomness.However, it should be noted that this module does not generate deterministic random numbers, This means that the sequence generated can not be repeated using some seed value.
- **matplotlib :-**This module has been used for plotting the graphs required in the assignment.
- **scipy.stats.chisquare :-** This module is used to perform the chisquare test for independence.
- **scipy.stats.pearsonr :-** This module is used to calculate the pearsons correlation coefficie.nt
- **scipy.stats.spearmanr :-** This module is used to calculate the spearmans correlation coefficient.
- **scipy.spatial.voronoi_2d :-** This module assists in plotting the voronoi plots which in turn assist in determining randomness of generated points.

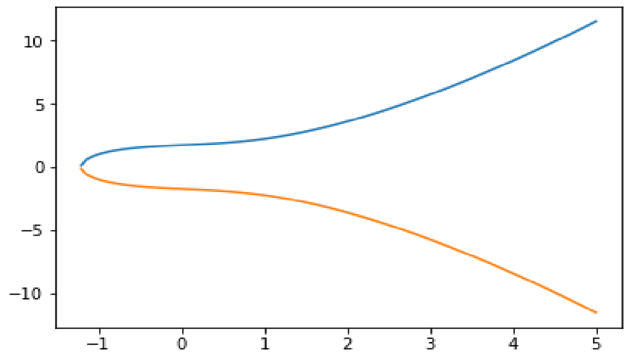

Fig. 3. Elliptic Curve over the field of real numbers with the chosen parameters i.e a=1, b=3, p=15733

V. RESULTS

The first step was to do an exhaustive search for a point P such that the order of P was a large(greater than 5000) prime. Our exhaustive search experiments show that such a point is :-
X=0, Y=4124 with and order(k) of **5281**.
Using k = $\lceil \mathbf{log_2\ q} \rceil$ we get the value of k as 12. Following this, algorithm 5 was used to generate all possible binary strings of length 12 and then using a cryptographically secure random number generator we select 5000 random strings(each with a decimal value lesser than order of P i.e. 5281) out of the 2^k possible binary strings. Using these 5000 binary strings, we compute 5000 points belonging to the cyclic group generated by the point P.

Analysis of the results :-

1) *Testing whether the Q values are uniform-randomly distributed on the Cyclic Group generated by P*

We have determined above that P is a point with an order of 5281. In order to generate a large number of Q values, we first generate a large number of random integers. We do this by randomly selecting 5000 k bit strings.This in turn is done by generating 5000 random numbers(all lesser than order of Point P) using the secrets module in Python which generates cryptographically secure random numbers. Thus, we can be sure that these numbers follow a uniform-random distribution. We can verify the uniform randomness by plotting a histogram for the values generated. The **histogram** appears as in Fig. 4 :-

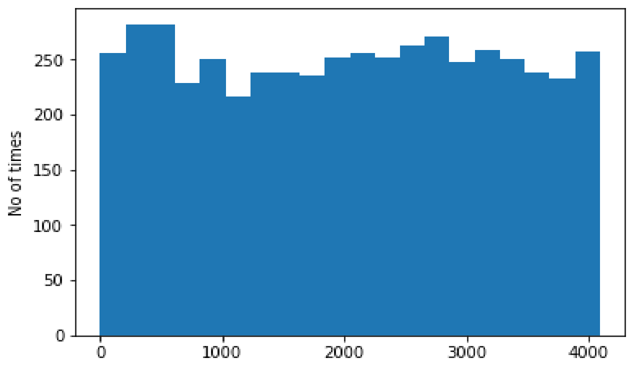

Fig. 4. Histogram with bin size 20

From Fig. 4, we can see that the distribution of the numbers is more or less uniform with some randomness added to the distribution.

We have further used the **chi-square test for independence** to determine whether each of the values are independently distributed or not. In order to perform the chi square test for independence, we first prepare a dictionary of the generated numbers with the number as the key and the frequency of the numbers as the value. Then we test for independence between the numbers by calculating the p-value and comparing it with the value of the significance level(0.05). *p-value* $\leq \alpha$ implies that the variables have a statistically significant association (Reject H_0) *p-value* $> \alpha$ implies that we cannot conclude that the variables are associated (Fail to reject H_0) Here H_0 is the null hypothesis which states that the variables are independent of each other. Since in our case p-value evaluates to be 0.60488 which is greater than 0.05, we fail to reject H_0. It should be noted that while independence does not imply randomness, the absence of independence is not a suitable condition for randomness.

Since all the decimal values of the randomly chosen strings are lesser than the order of P and they have a uniform random distribution, we can also say that the points generated will have the same distribution as the random numbers generated above(because each decimal value maps to a unique point in the cyclic group generated by P).

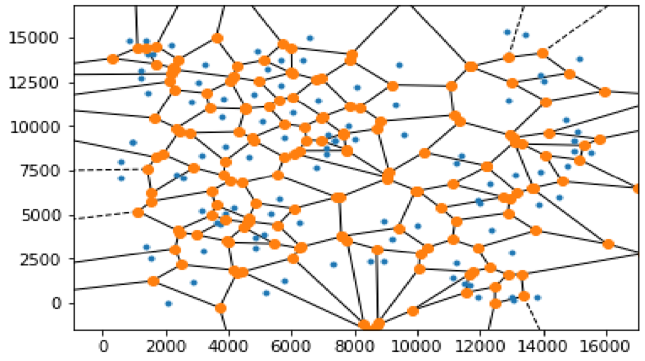

Fig. 5. Voronoi Plot for Q values generated

The randomness can also be observed in terms of the **Voronoi Diagrams**(Fig. 5).The Voronoi diagram partitions R^2 into regions based on the samples. Each sample *x* has an associated Voronoi region *Vor(x)*. For any point y $\in$ *Vor(x)*, *x* is the closest sample to *y* using Euclidean distance. The different sizes and shapes of these regions in Fig. 5 gives some indication of the randomness of the points obtained.

2) *Correlation between Q values and k-bit input binary strings :-*

The correlation has been tested in the following three ways :-

- We select the binary strings in such a way that they all have a high correlation coeficient(both Pearson and Spearman) with respect to each other i.e. correlation coefficient should be greater than 0.5. The decimal values of the strings selected are :- 11,15,31,47,79. The binary strings corresponding to these numbers are :-

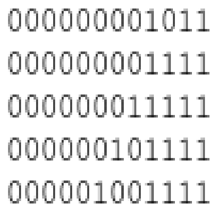
000000001011
000000001111
000000011111
000000101111
000001001111

Fig. 6. Binary Strings with high statistical correlation

The plot of the points generated corresponding to these strings can be seen in Fig. 7.

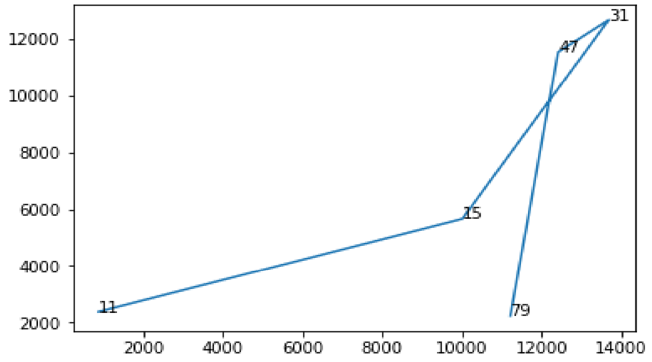

Fig. 7. Points corresponding to the binary strings above

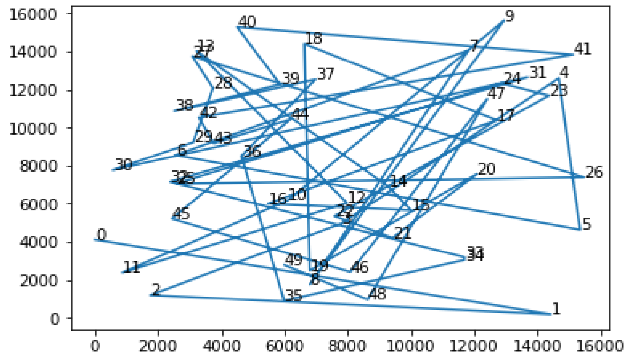

Fig. 8. Points labelled with the decimal values that generate them

From Fig. 7, it can be concluded that even though the strings have a high statistical correlation, there is no identifiable pattern between the points generated using Q = d.P

- We use the Pearsons Correlation Coefficient to estimate the correlation between the decimal values(d) from 1 to 1000 and the points generated using Q = d.P. We calculate the coefficient for x coordinates and the y coordinates of the points separately. The values of Pearson coefficients are :-
  - -0.0385
  - -0.0455

  for x coordinates and y coordinates respectively. These values show that there is almost no correlation between the decimal values and the points obtained on the Elliptic Curve under modulo p corresponding to these decimal values.
- We also use the Spearmans Correlation Coefficient to estimate the correlation between the decimal values(d) from 1 to 1000 and the points generated using Q = d.P. We calculate the coefficient for x coordinates and the y coordinates of the points separately. The values of Spearman coefficients are :-
  - -0.03715
  - -0.04596

  for x coordinates and y coordinates respectively. These values show that there is almost no correlation between the decimal values and the points obtained on the Elliptic Curve under modulo p corresponding to these decimal values.

VI. DISCUSSION

In this paper we have not used one of the cryptographically secure elliptic curves available in the market for experimentation because these secure curves have generators with extremely large orders which would've made it almost impossible to perform the listed tasks on an ordinary laptop. As a result, the term "large" here has been defined as any value greater than 5000 for the scope of this paper. The secrets module has been used to generate random numbers as the numbers generated by the random module offered by Python are not cryptographically secure.It should however be noted that while most random number generators used in Cryptography are deterministic in nature, the secrets module does not exhibit determinism. This however is not a problem for us as we merely use it once to select random strings from a larger set of strings and do not require determinism at all. In this paper, the Pearsons Coefficient and Spearmans Coefficient have been used to calculate the statistical correlations. The formula for Pearsons coefficient is

$$r = \frac{N\sum xy - (\sum x \sum y)}{\sqrt{[N\sum x^2 - (\sum x)^2][N\sum y^2 - (\sum y)^2]}} \tag{13}$$

where N is the number of samples and x and y represent the values of the variables between which the correlation is being calculated. Correspondingly, the formula for Spearmans Coefficient is :-

$$rho = \frac{6\sum d_i^2}{n(n^2-1)} \tag{14}$$

where d = the pairwise distances of the ranks of the variables x_i and y_i and
n = the number of samples.
The reason we have used the two coefficients is that the Pearsons coefficient only measures linear trend(correlation) while the Spearman's correlation measures increasing or decreasing trend(correlation) that is not necessarily linear.
We have also used the chi square test for independence to determine the independence of the numbers generated by the random number generator.
It should be noted that while independence does not imply randomness, the absence of independence is not a suitable condition for randomness.
The Voronoi plots have been used as they give us a means to visualise the randomness in a two dimensional space and the histograms give us a means to visualise the uniform nature of a distribution.

VII. CONCLUSION

From the above experiments, we can conclude several important aspects about Elliptic Curve Cryptography.
If we plot an elliptic curve over Z_p , we do not get anything remotely resembling a curve. For an Elliptic Curve to be cryptographically secure, the ECDLP should be hard for a given elliptic curve. For the ECDLP problem to be hard, we need a large enough Cyclic Group on the Elliptic Curve over Z_p. The security of the curve largely depends on the number of points on an eliptic curve over the field Z_p which should ideally be a large prime number so as to avoid a small subgroup attack.
The fact that there is almost no correlation(linear or otherwise) between the values of d and Q in Q=d.P, also ensures the difficulty of the ECDLP. Thus, given Q and P, it is impossible to determine d other than by using the brute force method(which is unfeasible when the order of the chosen point is very large).This characteristic of Elliptic Curves facilitates their use in Public Key Cryptography where Q is usually the public key and d is the private key.

REFERENCES

[1] https://realpython.com/python-histograms/
[2] https://www.geeksforgeeks.org/find-square-root-modulo-p-set-2-shanks-tonelli-algorithm/
[3] https://stackoverflow.com/questions/3949226/calculating-pearson-correlation-and-significance-in-python
[4] Cryptography and Network Security by Forouzan and Mukhopadhyay
[5] Understanding Cryptography by Christoff Paar
[6] https://docs.python.org/3/library/secrets.html
[7] https://pynative.com/python-secrets-module/
[8] https://crypto.stackexchange.com/questions/44337/security-of-elliptic-curves
[9] https://www.geeksforgeeks.org/modular-exponentiation-power-in-modular-arithmetic/
[10] https://en.wikipedia.org/wiki/Tonelli%E2%80%93Shanks_algorithm
[11] https://www.cse.wustl.edu/ jain/cse567-08/ftp/k_27trg.pdf
[12] https://support.minitab.com/en-us/minitab/18/help-and-how-to/statistics/tables/how-to/chi-square-test-for-association/interpret-the-results/key-results/
[13] http://planning.cs.uiuc.edu/node201.html
[14] https://crypto.stackexchange.com/questions/66441/why-only-non-prime-order-fields-have-small-subgroup-attacks
[15] https://stackoverflow.com/questions/47514695/whats-the-difference-between-os-urandom-and-random
[16] http://web.math.rochester.edu/people/faculty/doug/mypapers/wayne1.pdf
[17] https://www.quora.com/What-is-the-difference-between-dependence-and-correlation