Advances and Applications in Mathematical Sciences
Volume 21, Issue 9, July 2022, Pages 5445-5456

A CONVERSION ALGORITHM OF TEXT MESSAGES INTO ELLIPTIC CURVE POINTS

TAPAS KUMAR GHOSH

Department of Computer Science
Bankura Sammilani College
West Bengal, India
E-mail: tapas.bsc38@gmail.com

Abstract

In our everyday life use of Internet is increasing rapidly which results heavy traffic in the Internet. Therefore for congestion control a common goal is to reduce number of bits transmitted through the Internet. Sensitive data transmitted through the Internet by encrypting them. Various encryption algorithms are available to transmit sensitive data through the Internet. Elliptic Curve Cryptosystem is a recent one of them. In Elliptic Curve Cryptosystem messages are generally converted into Elliptic Curve points. In this paper we have suggested a new method of mapping text messages in Elliptic Curve points which reduces the number of bits remarkably without compromising the security.

I. Introduction

Use of Internet is increasing rapidly in our day-to-day life. According to Cisco [1] the worldwide total internet traffic has increased 3.2 times from 2016 to 2021 with a compound growth rate of 26% per annum. In this connection a major challenge is, to reduce the number of bits to be transmitted through the Internet. Now a day several exabytes of sensitive data transferred through the Internet. Sensitive data transmitted through the Internet in encrypted form. Among several encryption algorithms Elliptic Curve Cryptography is a recent one. In 1985 Neil Koblitz [2] and Victor Miller [3] were the first who independently proposed Elliptic Curve cryptosystem. Initially this cryptosystem was not so popular. But with time popularity of ECC is increasing gradually. In ECC much smaller key size

2020 Mathematics Subject Classification: 11G05.

Keywords: Elliptic Curve, Encoding, Decoding, Plaintext, Ciphertext.

Received December 3, 2021; Accepted February 28, 2022

than the key size in RSA is required, which reduces the processing complexity [4]. This feature makes ECC suitable for use in devices like mobile phones, smart cards, IoT applications etc, where power consumption and memory space availability is constrained [5]. In every cryptographic system a method is needed to convert plaintext characters or symbols into a numerical value which can be used to do mathematical computation. Several researchers have searched for a mapping method of messages into Elliptic Curve points [2, 5, 6, 7, 8]. Among all these methods a suitable approach is to write ASCII values of the plaintext characters side by side and use the numeric value for computation of ciphertext. Since each ASCII character is of seven bits long so a highly frequent character takes space as long as a rarely frequent character. In English literature few characters are used frequently all others are used relatively infrequently. So if we use very small values for frequently used characters and high values for rarely used characters then size of the encrypted text files can be reduced. In this paper a new algorithm with this approach is proposed, to encode text messages into elliptic curve points. For long messages, comparing with writing ASCII values side by side this new algorithm reduces number of bits used in the generated points. The points obtained thus are then used to generate other points on the same Elliptic Curve by the El Gammal algorithm or Massey-Omura algorithm [9]. These newly generated points can then be used as ciphertext. After this brief discussion the remaining parts of this paper is organized as follows: In Section II a brief description of elliptic curve cryptography over a prime field GF (p) and few mapping methods are discussed. The proposed scheme of encoding simple text message into EC points, and decoding from EC points to original message is discussed in section III. In section IV a comparison of number of bits, required for the proposed scheme and other method is discussed. Section V concludes the paper.

II. Related Work

The simplified Weierstrass equation of an elliptic curve over a prime field GF (p) is

$$E : y^2 = x^3 + ax + b (\text{mod } p) \tag{1}$$

Where $a, b \in Zp$ and $\Delta = 4a^3 + 27b^2 \neq 0 \bmod p$.

For a point P on E, nP operation (multiplying an integer value n with P) can be computed easily with add double rule [2]. For large integer value of n computation of nP is easy but it is very hard to find out n from $Q = nP$ if P, Q are given. This is known as the Elliptic Curve discrete logarithm problem. Let $A(x_1, y_1)$ and $B(x_2, y_2)$ are two points on E. $A + B$ and can be computed using the following formulae.

$$\alpha = \{\frac{y_2 - y_1}{x_2 - x_1} \bmod p; \text{ if } P \neq Q \text{ (For addition of points)} \tag{2}$$

$$\alpha = \frac{3x_1^2 + a}{2y_1} \bmod p; \text{ if } P = Q \text{ (For doubling a point)} \tag{3}$$

Obtain α using (2) for $A + B$ or (3) for $2A$. We can get the resulting point (x_3, y_3) as

$$x_3 = (\alpha^2 - x_1 - x_2) \bmod p$$

$$y_3 = (\alpha(x_1 - x_3) - y_1) \bmod p$$

The points on E forma finite abelian group with identity element as the point at infinity. Using Massey-Omuraalgorihm or ElGamal algorithm communicating parties can exchange data secretly. For implementation details of these algorithms one can see [9, 10].

Both of these algorithms use points on E for encryption and decryption purpose. So a mapping algorithm to convert the plaintext message into points on an elliptic curve is necessary. Numerous mapping algorithms for mapping messages into elliptic curve points are discussed by researchers. Few important ones of those are discussed below.

Mapping Method 1: In [2] authors had discussed this method. They first had taken a base point P on the Elliptic Curve E where messages are to be mapped. ASCII value of each character of the message was multiplied with P. for example ASCII value of c is 99. So the mapping point of c on E is 99*P. this is the easiest method of mapping messages into Elliptic Curve points. But secure curves in today's standard with this method will generate

enormous number of bits. Also this method is vulnerable to statistical attack.

Mapping Method 2: This method [7] is an extension of method-1. Here authors used a matrix to permute the generated points so that frequency analysis attack becomes infeasible. But since the basic approach (i.e. multiplying base point with character's ASCII value) is same with method -1 then this algorithm also will generate huge number of bits.

Mapping Method 3: In this method [6] 8 bit extended ASCII values of characters are written side by side and converted into Elliptic Curve points. This method reduces the number of bits than Method-1 and Method-2. Instead of using 8 bit extended ASCII values, in the result section proposed method is compared with this method but writing 7 bit ASCII values side by side.

III Proposed Mapping Method

In our proposed method in lieu of ASCII values a prime number is used as value of each character. For a smaller set of prime values we assign same prime number to mean both uppercase and lowercase of a letter. The prime numbers are assigned to each character in such a way that lowest frequent character has the highest prime value and vice versa. The frequency of letters in English literature is assumed from [11]. For punctuation marks, digits and other symbols an analysis is made from different books, research articles set of messages received through e-mail etc. From the above mentioned analysis and from [11] we have prepared the table of assigned values of each character as follows:

Table 1.

Sl. No	Character	Assigned Value	Sl. No	Character	Assigned Value
1	Space	2	35	5	149
2	E / e	3	36	6	151
3	T / t	5	37	7	157
4	A / a	7	38	8	163
5	O / o	11	39	9	167

6	I / i	13	40	“	173
7	N / n	17	41	/	179
8	S / s	19	42	;	181
9	H / h	23	43	:	191
10	R / r	29	44	!	193
11	D / d	31	45	?	197
12	L / l	37	46	‘	199
13	C / c	41	47	+	211
14	U / u	43	48	_	223
15	M / m	47	49	-	227
16	W / w	53	50	(	229
17	F / f	59	51	)	233
18	G / g	61	52	{	239
19	Y / y	67	53	}	241
20	P / p	71	54	[	251
21	.	73	55	]	257
22	B / b	79	56	=	263
23	V / v	83	57	%	269
24	K / k	89	58	#	271
25	J / j	97	59	@	277
26	X / x	101	60	$	281
27	Q / q	103	61	<	283
28	Z / z	107	62	>	293
29	,	109	63	\	307
30	0	113	64	`	311
31	1	127	65	*	313

32	2	131	66	~	317
33	3	137	67	^	331
34	4	139	68	|	337

Since communicating parties should agree with character values so table 1 is a public file. Before running suitable encryption algorithm we create a set of numbers U from plaintext. The plaintext is scanned from left to right, anda set of characters is prepared from the plaintext, taking occurrence of each character or symbol exactly once. Let us say it as S. According to table-1 maximum length of S is 68. Consulting Table-1 values of characters in S are multiplied together to get a numeric value N. N should be computed in such a way, that it is less than $p/10$, where p is the public modulus of the chosen Elliptic curve on which plaintext is to be mapped. For a large set of characters the restriction $N < p/10$ may yield several values of N. Keep all these computed N's into U. Here $p/10$ operation is necessary to keep generated values sufficiently less than p, so that later we can multiply them with 10 and get values less than p.

Now a scheme is necessary to construct the plaintext from characters in S. Sort S in ascending order, according to assigned values of characters. Take position value of each plaintext character in this sorted sequence. Writing these position values side by side we will try to obtain numeric values less than $p/10$ and append them in U. Increase all numbers stored in U multiplying 10 with each of them. Taking these values as x co-ordinate we will try to solve equation-1 for corresponding y value failing which we increase x by 1 and again try to get corresponding y. this process is repeated maximum 10 times or less to get a point on (1). Encrypting these points by elliptic curve equivalent of El Gammal or Massey Omura algorithm we can now send the message. In the receiving end after decryption, from x co-ordinate of the points we will be able to get the characters forming the plaintext and position values of plaintext characters in set S. Using these positions again we will be able to reconstruct the original plaintext. In [13] we have used group of 9 characters to get each position uniquely. But this grouping yields very small number. To group more than 9 characters we should use position values to start from 10 and ended with 77. Therefore at

most 68(number of two digit numbers) characters can be used in a group provided we have a sufficiently large prime p. The algorithm for encoding a text message into points on an Elliptic curve over the prime field GF (p) is given below.

Output: Set of points on an Elliptic Curve $E; y^2 = x^3 + ax + b$ over GF (p); where p is a large prime.

Terminology:# is used to mean, number of elements in a list or a set. A[i] means, i-th element of set A or list A.

Step 1. Read the input text from left and create the set of characters (say S) used to form it taking each character exactly one time. Create an empty list of values. Say it as points.

Step 2. Sort S according to its character values as given in table-1. Say it S1.

Step 3. Set product=1and count=1. Repeat steps a and b, #S1 times.

a. If product*S1 [count] is less than p/10 then

product ← product*S1[count]

count ← count+1

b. else

Append product into points.

Set product ← 1.

product ← product*S1 [count]

count← count+1

Step 4: if product ≠ 1append product into points.

Step 5: Insert #points in beginning of points.

Step 6: Start reading text again from left. Set map to 0. Repeat steps a to c,#text times.

a. Find position value of read out character in S1. Say it x.

b. x ← x+9

c. if (map*100+x) less than p/10

map ← map*100+x

else

Append map into points.

map ← 0

map ← map*100+x

Step 7: Multiply each numeric value in points with 10.

Step 8: Considering each numeric value in points as x co-ordinate Use Koblitz's method to find out a point (x, y) on E.

Generated points can now be used for encryption using any Elliptic Curve analog of encryption algorithm. In the receiving end after decryption we can get back these points again. The message reconstruction algorithm from these points is given below.

Input: Ordered list of points on an Elliptic Curve $E; y^2 = x^3 + ax + b$ over GF (p); where p is a large prime.

Output: Plaintext message.

Step 1. Create an ordered list of numbers from input points taking only x values. Say it L.

Step 2. Divide each value in L by 10. Make only integer division. Replace old values in L with these newly computed values.

Step 3. Let the first number in L is n. From second position take first n values of L. Factorize these n values into prime numbers. Create a sorted list S of these prime numbers. Consulting table-1 replace each value in S with its corresponding character. Also create a null string S1.

// Now S consists of all characters which are used to create the plaintext.

Step 4. Set count to $n + 2$

Step 5. Do steps a and b, #L-(n+1) times.

a. x ← L[count]

I. Take next pair of digits from x (start from leftmost position). Assign it to y.

II. y ← y-9

III. Concatenate S[y] with S1 placing S[y] in right of S[1] .

IV. Repeat steps I to IV until x exhausts.

b. count ← count+1

After completion of running this algorithm we will be able to get back the original plaintext message.

Example and Illustration:

Below we explain our algorithm with a small example. Let us consider the group of words "Elliptic Curve Cryptography". This is a 27 characters string. The string is formed with uppercase and lowercase characters from {e, l, i, p, t, c, u, r, v, y, o, g, a, h}. Table-2 gives frequency and assigned prime values of characters of the string.

Table 2.

Character	Space	e	l	i	p	t	c	u	r	v	y	o	g	a	h
Prime Value	2	3	37	13	71	5	41	43	29	83	67	11	61	7	23
Frequency	2	2	2	2	3	2	3	1	3	1	2	1	1	1	1

We have used the prime modulus p from NIST recommended Elliptic Curve P-192 [12]. The product of the prime values of plaintext characters is 31468517337463416210. The character map is11181815231215191019201724 11101917222312142117132316 22. We can send these two values first by encoding them into points on the curve P-192, then by encrypting. In receiving end after decryption we get 31468517337463416210 and 11181815 2 31215191019201724111019172223121421171323162 2 again. From these values after factorizing first value we will be able to get the constituent characters. The sorted sequence (according to assigned prime values) of characters is (Space, 'e', 't', 'a', 'o', 'i', 'h', 'r', 'l', 'c', 'u', 'g', 'y', 'p', 'v'). From the character map now extract every pair of digits (starting from the MSB position) which, after subtracting 9 from each, represent the position of each

character in the sorted sequence. For example the fourth pair of digit is 15. Subtracting 9 we get 6. Sixth character in the sorted sequence is 'i' which has fourth position in the plaintext. Thus we can easily reconstruct the original message from the character map and the product of prime values of each character.

IV. Result and Discussion

Required number of bits is calculated using NIST recommended Elliptic Curve [12] as given below.

$$E : y^2 = x^3 + ax + b(\text{mod } p) \tag{4}$$

Where $a = -3$,

$b = 24551555460089438177402939151974517847691080581611912380 65$,

$p = 6277101735386680763835789423207666416083908700390324961279$.

All calculations are done using Magma programs [14]. As plaintext message we have taken abstract part of this paper. Result is summarized in table-3.Elliptic Curve is symmetric about x-axis. So every x-value will give two y values. Therefore for uniqueness of the result we have calculated number of bits only for xco ordinate of a point on the curve given in (4).

Table 3.

Message Mapping Method	Number of bits required (For x values only)	Number of generated points
Proposed Scheme	5417	30
If characters are written side by side using their ASCII values	7125	38

V. Conclusion

This paper proposes a new algorithm for conversion of text messages into Elliptic Curve points. The algorithm is developed for a sufficiently large set of 68 characters that are generally used for exchange of text messages, but it can be expanded up to a set of 90 different characters. This algorithm works efficiently for every secret communication where the maximum number of different symbols used is 90 and some of them have high frequency with all other have a relatively lower frequency. Encryptions of generated points hide all information about plaintext. Therefore an eavesdropper is unable to get plaintext information from ciphertext. So it is a secure algorithm. However a rigorous checking is necessary before its practical use.

References

[1] https://www.cisco.com/c/dam/m/en_us/solutions/service-provider/vni-forecast-highlights/pdf/ Global_2021_Forecast_Highlights.pdf

[2] Neal Koblitz, Elliptic Curve Cryptosystems, Mathematics of Computation 48 (1987), 203-209.

[3] Victor Miller, Uses of Elliptic Curve in Cryptography, Advances in cryptology-CRYPTO Springer Heidelberg 85(218) (1986), 417-426.

[4] Lap-Piu Lee and Kwok-Wo Wong, A Random Number Generator Based on Elliptic Curve Operations, Computers and Mathematics with Applications 47 (2004), 217-226.

[5] Omar Reyad, Text Message Encoding Based on Elliptic Curve Cryptography and a Mapping Methodology, Information Sciences Letters, 7, No 1, January 2018.http://dx.doi.org/10.12785/isl/070102

[6] Aritro Sengupta and Utpal Kumar Ray, Message mapping and reverse mapping in elliptic curve cryptosystem, Security Comm. Networks 2016; 9:5363:5375, DOI: 10.1002/sec.1702.

[7] F. Amounas and E. H. EI Kinani, Fast mapping method based on matrix approach for Elliptic Curve cryptography, International Journal of Information and Network Security 1 (2012), 54-59.

[8] Bh. Padma, D. Chandravati and P. Prapoorna Roja, Encoding and decoding of a message in the implementation of Elliptic Curve Cryptography using Koblitz's method, International Journal on Computer Science and Engineering 2(5), (2010), 1904-1907.

[9] Darrel Hankerson, Alfred Menezes, Scott Vanstone, Guide to Elliptic Curve Cryptography, Springer Professional Computing, 2004.

[10] Alfred J. Menezes, Paul C. Van Oorschot and Scott A Vanstone, A Handbook of applied Cryptography, CRC Press.

[11] https://en.wikipedia.org/wiki/Letter_frequency

[12] FIPS Publication 186-4, July 19, 2013.

[13] T. K. Ghosh, A Method of Text Message Mapping in Elliptic Curve Cryptosystems, International Journal of Computer Sciences and Engineering 7(3) March 2019.

[14] John Cannon, WiebBosma, Claus Fieker, Allan Steel (Editors), Handbook of Magma Functions, Version 2.19.