I. J. Mathematical Sciences and Computing, **2023, 2, 1-9**
Published Online on May 8, 2023 by MECS Press (http://www.mecs-press.org/)
DOI: 10.5815/ijmsc.2023.02.01

An Improved Security Schematic based on Coordinate Transformation

Awnon Bhowmik*
Department of Mathematics, University of Central Florida
E-mail: awnonbhowmik@outlook.com
ORCID iD: https://orcid.org/0000-0001-5858-5417
*Corresponding Author

Mahmudul Hasan
Department of Mathematics, University of Dhaka
E-mail: mahmudul-2016713604@math.du.ac.bd

Received: 05 February, 2023; Revised: 05 March, 2023; Accepted: 10 April, 2023; Published: 08 May, 2023

Abstract: An earlier research project that dealt with converting ASCII codes into 2D Cartesian coordinates and then applying translation and rotation transformations to construct an encryption system, is improved by this study. Here, we present a variation of the Cantor Pairing Function to convert ASCII values into distinctive 2D Coordinates. Then, we apply some novel methods to jumble the ciphertext generated as a result of the transformations. We suggest numerous improvements to the earlier research via simple tweaks in the existing code and by introducing a novel key generation protocol that generates an infinite integral key space with no decryption failures. The only way to break this protocol with no prior information would be brute force attack. With the help of elementary combinatorics and probability topics, we prove that this encryption protocol is seemingly infeasible to overcome by an unwelcome adversary.

Index Terms: coordinates, transformations, improvements, ciphertext

1. Introduction

New cryptosystems have increasingly become more obscure over time, increasing confusion and dispersion. Any cryptosystem's core relies on a unique mathematical trapdoor function that makes it nearly hard for an unauthorized interceptor to access sensitive data. Due to this, better security protocols are now being employed in commonplace digital systems. The remarkable fact is that, over time, it has become apparent that, with the proper application, any mathematical concept may be used to generate trapdoors for cryptosystems. In an earlier paper discussed here [1], a simple encoding system was discussed that primarily relied on transforming a message to two-dimensional coordinates followed by shifting and rotating them.

Geometric coordinates have been used with encryption before. Numerous previous studies on coordinates are actually still accessible, but they mainly cover topics like Elliptic curves [2], binary fields [3], finite fields [4], isogeny [5], Montgomery coordinate [6], etc. Point addition is used in elliptic curve cryptography. The computation of modular inverses for the constituents of the underlying finite field is required for point addition in affine coordinates. The extended Euclidean method can perform modular inversion, but at a cost that is roughly 100 times more than that of a single addition or multiplication in the finite field. However, there is no need to calculate a modular inverse for point addition in projective and Jacobian coordinates [7]. In projective or Jacobian coordinates, it is therefore cheaper to add two elliptic curve points than it is to do so in affine representation. Our study offers readers a fresh perspective by focusing just on cartesian coordinates and showing how to apply them to mathematics that doesn't include any of the subjects. Despite being a recreational endeavor and not being designed with industrial standards in mind, this may, with the right tweaks, be fairly useful in modest-scale operations.

It is difficult in any communication system, including the internet, satellite, and mobile, to prevent eavesdropping or data loss when the information is disseminated across the channel. Information security is now more crucial than ever for any application [8]. In the complexity and security of the public key cryptosystem, prime integers are crucial. The security of the public key algorithm known as RSA depends on how challenging the integer factorization problem is. Any encryption scheme is dependent on the key's length and the computing work needed to decrypt it. Although this is still used in conjunction with other systems such as OpenSSL which provides hybrid encryption with AES+RSA, RSA+ECC, etc.

A high-performing cryptosystem is typically constructed in levels, with the first layer typically consisting of an asymmetric cryptosystem that permits communication between both parties through insecure channels. Asymmetric systems are not recommended for sharing large amounts of data since they are computationally expensive. As a result, the following layer is typically a symmetric system, which is computationally cheaper but necessitates that both parties possess the same private keys for data encryption and decryption. Asymmetric systems are used to exchange these private keys through insecure public channels.

In the previous research endeavor we introduced an encryption algorithm that deals with 4 parameters whose working domain for successful decryption was left unfinished. Furthermore, the code written for that specific algorithm had numerous delicate faults that contributed to an increase in the runtime complexity of the program. In this research paper, however, we have been able to increase the security of the previous work drastically by introducing 6 parameters, and have tested larger and larger working domains for 100 percent successful encryption and decryption cycles to be used in the program. We have also shown mathematically that the encryption protocol is now practically impossible to break for anyone that tries to intercept messages sent through this.

2. Confusion and Diffusion

In cryptography, confusion and diffusion are two properties of the operation of a secure cipher identified by Claude Shannon in his paper Communication Theory of Secrecy Systems, published in 1949 [9]. In Shannon's original definitions, confusion refers to making the relationship between the key and the ciphertext as complex and involved as possible [10]; diffusion refers to the property that the redundancy in the statistics of the plaintext is "dissipated" in the statistics of the ciphertext. In other words, the non-uniformity in the distribution of the individual letters (and pairs of neighboring letters) in the plaintext should be redistributed into the non-uniformity in the distribution of much larger structures of the ciphertext, which is much harder to detect. In diffusion, the statistical structure of the plaintext is dissipated into long-range statistics of the ciphertext. Diffusion means that the output bits should depend on the input bits in a very complex way. In a cipher with good diffusion, if one bit of the plaintext is changed, then the ciphertext should change completely, in an unpredictable or pseudorandom manner. AES protocol is a perfect example that exhibits both confusion and diffusion.

3. Improvised Pairing Method

Integers and rational numbers have the same cardinality as natural numbers, which is a surprising result of set theory. A common tactic created by Cantor called diagonal progression can be used to demonstrate this. The Cantor pairing function serves as the foundation. The Cantor Pairing Function is a primitive recursive pairing function $\pi: \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ defined by

$$\pi(k_1, k_2) = \frac{1}{2}(k_1 + k_2)(k_1 + k_2 + 1) + k_2 \tag{1}$$

It can also be extended to multiple dimensions by repeated compositions $\pi^{(n)}: \mathbb{N}^n \to \mathbb{N}$

$$\pi^{(n)}(k_1, k_2, \cdots, k_n) = \pi\left(\pi^{(n-1)}(k_1, k_2, \cdots, k_{n-1}), k_n\right) \tag{2}$$

The following outlines the steps of splitting a given natural number into two natural numbers with Cantor unpairing. To retrieve an ordered pair (x, y) from a given t, the following transformations are used.

$$\begin{gathered} \omega = x + y \\ t = \frac{1}{2}\omega(\omega + 1) \\ z = t + y \end{gathered} \tag{3}$$

From equation (3), cross multiplying gives a quadratic in ω.

$$\omega^2 + \omega - 2t = 0$$

Solving it gives us

$$\omega = \frac{\sqrt{8t + 1} - 1}{2}$$

Which is a strictly increasing and continuous function when t is non-negative real. Since

$$t \le z = t + y < t + +(\omega + 1) = \frac{(\omega + 1)^2 + (\omega + 1)}{2}$$

This implies that

$$\omega \le \frac{\sqrt{8z + 1} - 1}{2} < \omega + 1$$

And thus

$$\omega = \left\lfloor \frac{\sqrt{8z + 1} - 1}{2} \right\rfloor$$

This finally allows the calculation of x and y from z as follows.

$$\omega = \left\lfloor \frac{\sqrt{8z + 1} - 1}{2} \right\rfloor$$
$$t = \frac{\omega^2 + \omega}{2}$$
$$y = z - t$$
$$x = \omega - y$$

Since this function uniquely splits an integer into a two-dimensional Cartesian coordinate, if used directly in a cryptosystem, this makes it vulnerable to a brute force attack.

4. The Asymmetric RSA Layer

Data is often authenticated using digital signatures and asymmetric cryptography. A correspondence, piece of application, or digital document can have its integrity and validity verified using a digital signature, which is a mathematical process. It serves as a digital substitute for a handwritten or stamped seal. Digital signatures, which are based on asymmetric cryptography, can guarantee evidence of the origin, identity, and status of an electronic document, transaction, or message and recognize the signer's informed permission. Systems with several users who may need to encrypt and decode communications can also employ asymmetric cryptography, such as encrypted email, SSL/TLS, cryptocurrencies, etc. The pros and cons of an asymmetric system are outlined below:

Table 1. Difference between symmetric and asymmetric cryptography

Pros	Cons
There is no need to exchange keys, hence the issue with key distribution is solved.	Compared to symmetric cryptography, it is a slow procedure. Therefore, decrypting bulk communications is not a good use for it.
Since the private keys never need to be communicated or made public, security is enhanced.	An individual cannot decode the communications he receives if he misplaces his private key.
It is possible to employ digital signatures so that a recipient may confirm the identity of the sender of a communication.	Nobody can verify that a public key belongs to the person supplied since public keys aren't authenticated. Users must thus confirm that their public keys are theirs.
It enables nonrepudiation, which prevents the sender from retracting a communication.	A malevolent actor who discovers a person's private key has access to that person's messages.

Although it is simple to multiply two huge prime numbers, the security of public-key cryptography is based on the difficulty of factoring, which is the process of separating the original numbers from the product [11]. Most attackers are unable to factor the product of two sufficiently big primes because it takes too long. Governments and businesses are shifting to a minimum key length of 2048 bits or higher since RSA keys that were once 1024 or 2048 bits long were broken. In fact, certain personal computers have started using a default key length of 3072 bits for the inherent RSA layer.

Our unique symmetric system's private parameters are exchanged through this layer. In this situation, the party with an interest in decrypting ciphertexts generated by our system selects a private RSA key and publishes the corresponding public key. The person encrypting data using our method receives the associated ciphertext and randomly generated private parameters, which they can communicate through insecure public channels. A complex technique is used to process the secret parameters and combine them into a single string. This string must be encrypted using the decryptor's public RSA key before being transferred.

5. The Algorithm for Processing Private Parameters

5.1 Encryption

The encryptor generates the six private integer parameters from a selected random distribution. These private parameters are joined into a single string to form the shared secret key. The algorithm used during the encryption process to produce the shared key is given below:

1. An array (Parameter Array) is used to store all six parameters in a specific order, in our implementation this order is $h_1, k_1, h_2, k_2, n_1, n_2$.
2. An array (Sign Array) containing the digits 1 and 2 is formed based on the array of parameters, where 1 indicates a positive integer parameter and 2 indicates a negative integer parameter.
3. An array (Length Array) containing the length of each parameter is formed based on the array of parameters.
4. A single string containing six distinct parts containing information about the six parameters is formed by iterating over these three arrays simultaneously. At each iteration, a part of the string is obtained by joining the value stored in Sign Array, Length Array, and Parameter Array respectively.
5. This string is encrypted using the asymmetric layer and is then sent through a public channel along with encrypted data obtained from our system to the decryptor.

5.2 Decryption

The decryptor receives the encrypted shared key and decrypts it to obtain the string containing all private parameters. The private parameters are then extracted from the string as given below:

1. The string is converted into an array of integers (Shared Array).
2. A loop is used to extract values of the Length Array and their corresponding indices in the shared array in two separate arrays.
3. As the indices containing the sign digits is one less than the indices of the length digits in the Shared Array, the indices of the sign digits in the shared array and corresponding values are easily obtained and stored in two separate arrays.
4. Lastly the information contained in these four arrays is combined to obtain the actual value of the secret parameters from the Shared Array.
5. Then these parameter values are used to decrypt data that was encrypted using our system.

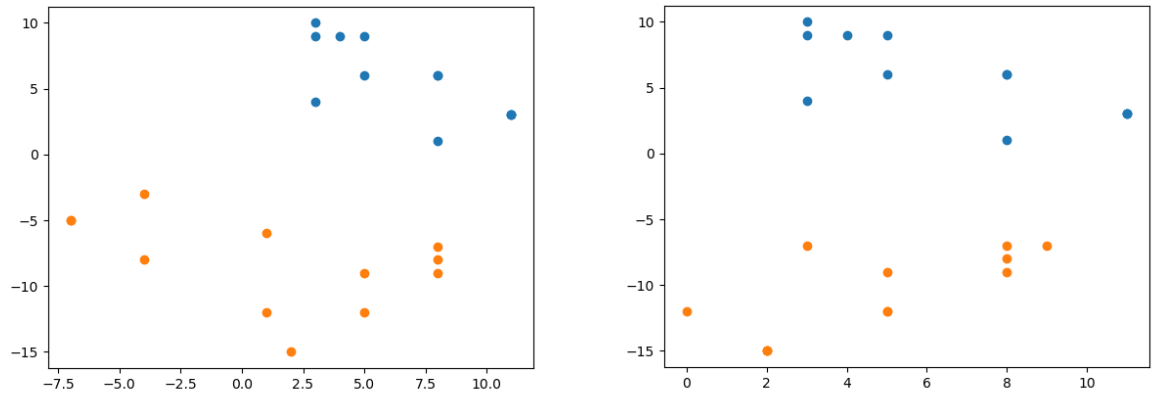
6. Encryption and Decryption Algorithms

6.1 Mapping ASCII values to 2D points

Previously Koblitz encoding and decoding algorithms were used to transform integer ASCII values into 2D points, in this paper we have instead used the Cantor Pairing Function for this particular purpose. One of the main reasons for this change is that Koblitz's encoding and decoding algorithms produced a significant amount of decryption failures. The highest observed successful decryption rate was 82%, which we thought was not acceptable for a proper cryptosystem. Using Cantor Function instead introduces some security issues, which stem from the fact that the Cantor Pairing Function uniquely maps an integer value to a point. Our rationale for this change is that using the Cantor Pairing Function will allow us to use more complex mathematical mappings to higher dimensions from integer values. This is also in accordance with our future plans of investigating this system using hypercomplex numbers [12].

6.2 Improvements in the Coordinate Transformation Phase

In the previous implementation, the coordinate transformation was carried out in a simplistic manner. This resulted in transformed points that had a rather predictable redistribution. The algorithm has been improved by introducing parameter cycling for coordinate transformations for successive points, which introduces better security and a less predictable distribution of transformed points in the 2D plane. Furthermore, the unique mapping of the Cantor Pairing Function is masked due to this process and the same ASCII values that had been mapped to the same 2D point are mapped to different points at the end of the transformation phase. This makes identifying the original state of the point significantly harder.



(a) With Cycling (b) Without Cycling

Fig. 1. Point Distribution

6.3 Encryption Algorithm

1. Six private parameters used in the transformation phase are chosen from a random distribution.
2. A plaintext string is separated into characters and stored in an array.
3. Every character in the array is then converted to its corresponding ASCII values.
4. Inverse Cantor Pairing is used to map each ASCII value to a 2D point and these points are stored as a 2D array.
5. The private parameters randomly generated in the beginning is used to perform translation and rotation transformations on each point generated through Inverse Cantor Pairing.
6. First translation transformation is carried out and translation parameters used are cycled for successive points.
7. After all points are translated they, are then rotated and the rotation parameter is similarly cycled for successive points.
8. The encryption algorithm then terminates by generating a 2D array of transformed points, which serves as the ciphertext of this system.

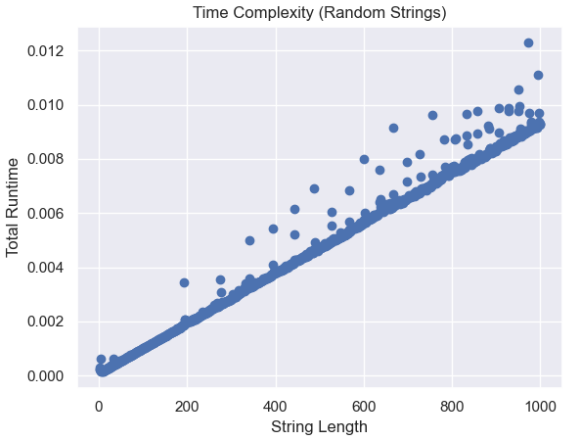
6.4 Decryption Algorithm

1. The private parameters used for coordinate transformation is extracted through the use of an asymmetric layer.
2. First the rotation of the points is reversed by using the same order of cycling of rotation parameter on successive points.
3. Then the translation is reversed by maintaining the same order cycling of translation parameters.
4. After the previous two steps the points are returned to their original state that was obtained using the Inverse Cantor Pairing Function.
5. The Cantor Pairing Function is then used to obtain the corresponding ASCII values for each 2D point.
6. The ASCII values are transformed into characters and merged into a single string to obtain the plaintext.

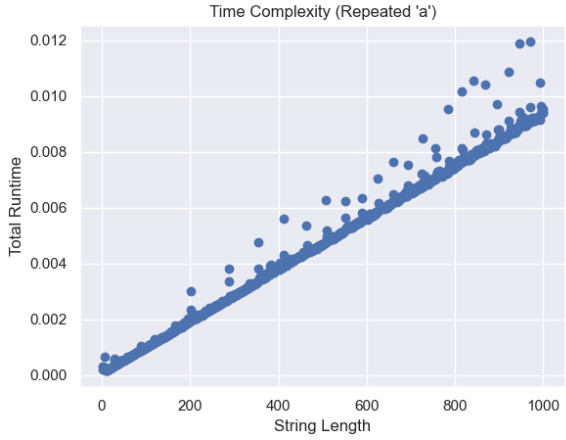
7. Time Complexity

The previous paper had numerous flaws in coding, such as calculating the inverse of a 2×2 square matrix inside a while loop while the cipher was decoded one letter at a time. This gave rise to an exponential time complexity overall. In this case, the inverse matrix formula has been hard coded into the system, and outside of any loops. Also, the previous paper involved sending over the private parameters to the decipherer, which is a wrong practice.

Due to the algorithms used being quite complex, it was not possible to theoretically determine the time complexity of the system. The empirical analysis provided below suggests that the algorithm for this system has a linear time complexity.



(b) String Length vs Time



(b) Repeated String vs Time

Fig. 2. Time Complexity Analysis

From the String Length vs Total Runtime plots, we can hypothesize that the time complexity of running the encryption and decryption algorithms together is approximately linear. Furthermore, the time complexity is not affected by the content of the string, which we can verify from the plots. The total runtime is similar even if the strings are made of random ASCII characters, or just formed from the repetition of the character 'a'. This is an expected outcome as the computation required for encrypting and decrypting all ASCII characters should be almost equal based on the algorithms used.

8. Key Space

After the key has been identified, it is equally vital to characterize it, which requires a thorough examination of the key space. The quantity of encryption/decryption key pairs that are available within the cryptosystem is known as the key space size. Let's define a key as k and a finite set of potential keys as K. This is also known as the key space and can be written as

$$K = \{k_1, k_2, \cdots, k_n\}.$$

The key in traditional cryptographic methods is often a string of randomly generated bits produced by some mechanical mechanism. These techniques are mostly based on number theory. Every conceivable n-bit key in such a procedure must be equally likely, with probability 2^{-n}, if the key is n bits long [13].

The key space can be made arbitrarily large as each parameter is chosen from the set of integers. Theoretically, this implies that the key space for this system is infinitely large. Even if we limit ourselves to choosing random integer parameters from a range of $[-10^{10}, 10^{10}]$ we get a key space containing $256 \times 10^{40} + 4$ unique keys. This is still a modest range for choosing integer parameters. We can easily use larger key spaces with negligible impact on the runtime of the encryption and decryption algorithms. Still, we provide simple mathematical proof of this as a lemma.

In general, keeping the hardware capacity of modern computers in mind, it is preferred that the key space will at least have a size $> 2^{100} = 10^{30}$. This size of key space is easily achieved if we set a lower bound for intervals to random integer parameters from $[-8 \times 10^6, 8 \times 10^6]$ which gives us a total of $(2 \times 8 \times 10^6 + 1)^4 \times 16$ possible unique keys.

Lemma 1. The key space generated by choosing each of the six integer parameters randomly from the set of integers is theoretically infinitely large.

Proof. Let us assume that the key space is not infinitely large. This implies that we can only assign finitely many values for each of the integer parameters, but this is a contradiction as the set of integers is infinite. Hence, the possible choices for each of the parameters are infinite.

As our key space is generated from unique combinations of integer parameters chosen from infinite sets, it follows that the key space generated will also have infinitely many unique keys.

9. Statistical Analysis

9.1 String Length vs Runtime

The following table shows the difference in runtime between the algorithms used in the previous paper and the current paper, depending on various string lengths. The improved algorithms appear to increase the overall performance

of the earlier version by about $15 - 20$ times. The use of the Cantor Pairing Function may be the core reason behind this performance boost, as we have made the algorithm for coordinate transformations considerably more complex than before by introducing cycling transformation parameters.

Table 2. Performance comparison in seconds

String Length	Previous Algorithm	Current Algorithm
26	0.00269	0.00077
260	0.02564	0.00336
2600	0.23829	0.02533
26000	2.31413	0.26784

10. Cryptanalysis

The private key parameters are chosen in an arbitrary range of $[-a, a]$. This means that the chance of an attacker guessing the correct parameter is $\frac{1}{2a+1}$. But there are 4 such parameters and 2 parameters related to rotation transformation that are interconnected and must be obtained simultaneously and exactly. This means that the overall probability of an interceptor cracking the system via brute force is

$$\frac{1}{(2a+1)^4 \times 4^2}$$

Since a can be arbitrarily large, it consequently implies that

$$\lim_{a \to \infty} \frac{1}{(2a+1)^4 \times 4^2} = 0$$

Table 3. Probability of calculating all parameters within different ranges

	Probability of guessing	
Range	1 Translation Parameter	4 Translation & 2 Rotation Parameters
$[-10^2, 10^2]$	0.004975124	$6.126546951 \times 10^{-10}$
$[-10^3, 10^3]$	0.00049975	$6.237515609 \times 10^{-14}$
$[-10^4, 10^4]$	4.99975×10^{-05}	$6.248750156 \times 10^{-18}$
$[-10^5, 10^5]$	4.99998×10^{-06}	$6.249875002 \times 10^{-22}$

Another major factor to consider here is trials. Let's say an interceptor tries to obtain the parameters via brute force, how many trial runs does he need to run, to guarantee that he hits the jackpot, that is, he hits the correct values for all parameters at least once? We are looking at the following

$$\begin{aligned} &n = ? \\ &p = \frac{1}{(2a+1)^4 \times 4^2}, \text{ for a range } [-a, a] \\ &q = 1 - p \\ &X \geq 1 \end{aligned}$$

Although, traditionally it is recommended that the keyspace has a size $> 10^{30}$, for which we need a greater than or approximately 8×10^6. It becomes computationally infeasible to compute the values shown in Table (\ref{tab:min_trial}) for such a value of a. So let us choose $a = 100$ for the sake of a numerical demonstration.

$$\begin{aligned} P[X \geq 1] &= 1 - P[X = 0] \\ &= 1 - \binom{n}{x} p^x q^{n-x} \\ &= 1 - \binom{n}{0} \left(1 - \frac{1}{201^4 \times 16}\right)^n \end{aligned}$$

These are the results we obtain for a $1\%, 10\%, 50\%$, and 90% chance of success.

Table 4. Number of trials for given success rates

Probability	Minimum Trials
1%	2.624730919×10^8
10%	2.751579720×10^9
50%	$1.810212975 \times 10^{10}$
100%	$6.013397338 \times 10^{10}$

11. Other Potential Uses of the Methods Used

This coordinate transformation process of this system may be used to introduce confusion in any encryption protocol that aims to encrypt images. In chaotic mapping-based encryption protocols, computational complexity may be reduced by implementing a coordinate transformation-based confusion-generating step instead of the classic Arnold's Cat Mapping and other mappings, which have the problem of being periodic.

Furthermore, although we have limited the use of this system to only ASCII values, it can be seamlessly implemented for Unicode characters without any modification.

12. Conclusion

In this paper, we have improved upon the previously implemented system in various ways. Although the practical usage of this cryptosystem may be limited, we believe that this system correlates with several interesting mathematical concepts. Even though the system shows promising results against brute force attacks, it may be possible to break the system using specific differential and statistical attacks based on the unique mapping of the Cantor Function. We aim to investigate these weaknesses in future studies of this family of cryptosystems based on coordinate transformations. It has also occurred to us during our investigation that coordinate transformations may have potential usage in Chaos Based Cryptosystems, where periodic chaotic mappings such as Arnold's Cat Mapping are used to introduce confusion in plaintexts before diffusing them [14, 15]. Using coordinate transformations in such cases may reduce computational complexity and also improve security by removing the periodicity that occurs from using such chaotic mappings, but we cannot be certain about these claims before investigating further. We also rearranged the structure of the system to use Cantor Function to enable further research related to higher dimensional mappings in the future and use more complex transformations.

All our work has been compiled into a repository which is available here [16].

References

[1] A. Bhowmik, "An encoding schematic based on coordinate transformations," International Journal of Mathematical Sciences and Computing(IJMSC), 2020.

[2] I. Setiadi, A. I. Kistijantoro and A. Miyaji, "Elliptic curve cryptography: Algorithms and implementation analysis over coordinate systems," in 2015 2nd International Conference on Advanced Informatics: Concepts, Theory and Applications (ICAICTA), 2015.

[3] D. Hankerson, J. López Hernandez and A. Menezes, "Software implementation of elliptic curve cryptography over binary fields," in International Workshop on Cryptographic Hardware and Embedded Systems, 2000.

[4] A. E. Cohen and K. K. Parhi, "Gpu accelerated elliptic curve cryptography in gf (2 m)," in 2010 53rd IEEE International Midwest Symposium on Circuits and Systems, 2010.

[5] L. De Feo, "Mathematics of isogeny based cryptography," arXiv preprint arXiv:1711.04062, vol. 12, 2017.

[6] T. Moriya, H. Onuki, Y. Aikawa and T. Takagi, "The Generalized Montgomery Coordinate: A New Computational Tool for Isogeny-based Cryptography," Cryptology ePrint Archive, 2022.

[7] M. S. Hossain, Y. Kong, E. Saeedi and N. C. Vayalil, "High-performance elliptic curve cryptography processor over NIST prime fields," IET Computers & Digital Techniques, vol. 11, p. 33–42, 2017.

[8] W. Diffie and M. E. Hellman, "Multiuser cryptographic techniques," in Proceedings of the June 7-10, 1976, national computer conference and exposition, 1976.

[9] C. E. Shannon, "Communication theory of secrecy systems," The Bell system technical journal, vol. 28, p. 656–715, 1949.

[10] W. Stallings, Cryptography and network security, 4/E, Pearson Education India, 2006.

[11] P. L. Montgomery, "A survey of modern integer factorization algorithms," CWI quarterly, vol. 7, p. 337–366, 1994.

[12] R. Voliansky, N. Volianska, V. Kuznetsov, M. Tryputen, A. Kuznetsova and M. Tryputen, "The Generalized Chaotic System in the Hyper-complex Form and Its Transformations," in International Symposium on Engineering and Manufacturing, 2022.

[13] G. Alvarez and S. Li, "Some basic cryptographic requirements for chaos-based cryptosystems," International journal of bifurcation and chaos, vol. 16, p. 2129–2151, 2006.

[14] S. Tedmori and N. Al-Najdawi, "Image cryptographic algorithm based on the Haar wavelet transform," Information Sciences, vol. 269, p. 21–34, 2014.

[15] M. Mahesh, D. Srinivasan, M. Kankanala and R. Amutha, "Image cryptography using discrete haar wavelet transform and arnold cat map," in 2015 International Conference on Communications and Signal Processing (ICCSP), 2015.

[16] A. Bhowmik and M. Hasan, CoordinateGeometryCrypto, 2022.

Authors' Profiles

Awnon Bhowmik was a former Ph.D. student at the University of Central Florida. He obtained his bachelor's from CUNY York College. His research interests are in Cryptography, Number Theory, Fractal Geometry, Mathematical Modelling, and Simulation. Currently working as a substitute teacher in the NYC Department of Education, he has worked as an educator for over 8 years.

Mahmudul Hasan is a final year undergraduate student of mathematics at the University of Dhaka. He has also taken online courses on Data Science from several online platforms. Professionally he has worked as a software engineer and web developer for numerous private institutions. He is also an educator who instructs students on programming and mathematics.

How to cite this paper: Awnon Bhowmik, Mahmudul Hasan, "An Improved Security Schematic based on Coordinate Transformation", International Journal of Mathematical Sciences and Computing(IJMSC), Vol.9, No.2, pp. 1-9, 2023. DOI: 10.5815/ijmsc.2023.02.01