Journal of Network and Information Security
9 (1) 2021, 01-05
http://www.publishingindia.com/jnis

High Speed Public Key Encryption using Hyperboloid for Work from Home

Kunal Gagneja
Faculty, BCA Department, Government College of Commerce & Business Administration, Chandigarh, India.
Email: kunal.gagneja@gmail.com

Abstract: The novel cryptosystem was designed to be fast and secure for cloud. Circle was used for encryption. A 3D hyperbola was used for decryption. Mathematical properties were used to ensure that an eavesdropper could not perform side-channel attacks. The number of equations which an eavesdropper had was lesser than the number of unknowns for every block. The cryptosystem could work anywhere from smartwatches to supercomputer. It was designed to be flexible. 3D hyperboloid cryptosystem could work as both stream and block cipher. There was no restriction on key size and block size. Confusion and diffusion were measured as parameters for security. It is a complete encryption type cryptosystem. Performance and security analysis was performed both in terms of hardware and software implementations. Problem of key management, handling and key distribution was solved. Experimental results indicate that the proposed cryptosystem was faster than both symmetric and asymmetric key cryptosystems.

Keywords: ECC, Public key cryptography, Quantum computer, Quantum encryption, RSA.

I. Introduction

Already existing public key cryptosystems [1] are either slow or insecure. It is because of their mathematical architecture that they become slow exponentially when plaintext data begins to grow in size. They are dependent on prime number, discrete logarithms and elliptic curves. These mathematical functions consume a lot of computational power both in hardware and software implementations. Moreover, increasing the block size makes the cipher secure but exponentially slow. The aim of creating a novel cryptosystem was to ensure that speed could be kept high even by increasing the block size many times. Elliptic curve cryptography has licensing restrictions. Moreover, various implementations of the same are not compatible with each other. They use third party for certification. Existing [2] public key cryptosystems are insecure against brute force attack by a quantum computer [3] [4]. Public and private keys are mathematically related. It is possible to calculate private key if public key and ciphertext are known [5] [6]. Public key cryptography is only applicable to small plaintexts. Many times, public key cryptography [7] is combined with hash function and symmetric key cryptosystems. PGP is secure but is mostly used for email encryption. It shows the need for a novel cryptosystem and corresponding processor. RSA was too old and insecure. Its derivatives have the same limitations. It was insecure against side-channel attacks. Even 4096 bit of RSA was broken by accoustic attack. PGP was designed for email encryptions. The derivatives of RSA have similar problems in their design. All the symmetric key cryptosystems lack authentication. Some cryptosystems can only do digital signature. So far, RSA is the only cryptosystem which can work both as block and stream cipher.

The problem was solved by two methods. First by increasing the speed. Second by increasing the randomness in ciphertext. Comparison of throughput was done with already existing public key cryptosystems [8]. Both stream and block ciphers were included too in experiments to prove that the proposed cryptosystem was the fastest. It was assumed that what was faster on classical hardware can possibly be faster on quantum hardware [9] [10] too. Mathematical analysis indicates that the design has got compression combined encryption. It was due to concepts of trigonometry. Reverse of the cryptosystem too can be done to provide digital signature [10]. Key size was kept larger for more security. Moreover, mathematical design allowed us. Security was dependent on the secrecy of the key and not on the algorithm. Error in one block was restricted to the same block only. Compression combined encryption also made network transmission easier. Experimental results indicated that padding was not required.

II. Proposed Solution

There are different ways of defining a hyperboloid. It can be described as collection of infinite circles with different radius but same centre in 3D. It can also be defined as collection of infinite hyperbolas with same foci. These points indicate infinite plaintext and ciphertext points. Let us assume a circle with centre (q, r). A 3D hyperbola has centre at (p, q, r). Centres of both were kept same to ensure faster decryption. Centres can be kept at different location in 3D too but this would unnecessarily slow the algorithm. The equation of circle was

$$q^2 + r^2 = Radius = Ciphertext \quad (1)$$

The equation of Hyperboloid was

$$1 = \frac{p^2}{l^2} + \frac{q^2}{m^2} - \frac{r^2}{n^2} \quad (2)$$

Plaintext was divided into two unequal parts namely q_1 and r_1. The common centre point namely (q, r) was used as public key. It was known to sender, receiver and eavesdropper. The private keys namely *p*, *l*, *m* and *n* were known only to receiver. First block of ciphertext was calculated using equation (3). It is an equation of circle in 2D with common centre as that of hyperboloid in 3D. It was decrypted using equations (3) and (4) at the receiver end.

$$(q - q_1)^2 + (r - r_1)^2 = Ciphertext \quad (3)$$

$$1 = \frac{(p - p_1)^2}{l^2} + \frac{(q - q_1)^2}{m^2} - \frac{(r - r_1)^2}{n^2} \quad (4)$$

The point (q_1, r_1) lies both on circle and hyperboloid. The receiver used equation (3) and (4) for decryption. The value of $(p - p_1)$ was calculated by sender using hyperbolic tan using equation (5). Only the sender knows about complete mathematical properties of hyperboloid. Sender uses properties of hyperbolic trigonometry to calculate the combined value of $(p - p_1)$. It changes after every block. Disclosing the same to receiver and eavesdropper on network was not a security flaw. The value of $(r - r_1)$ was already known to the sender. Note that *p* is a part of private key of sender and p_1 is temporary variable. Dimensions of hyperboloid were changed by randomly changing the value any one of *l*, *m* or *n* for every block. The new value of hyperbolic tan was used a public key be receiver. This changed the dimensions of hyperboloid It was required for security. It can be done with computational complexity of $O(n^2)$. The complexity of changing keys was the least as compared to other public key cryptosystems. The same design can be used to encrypt second block of plaintext.

$$\tan h = \frac{(p - p_1)}{(r - r_1)} \quad (5)$$

$$(q - q_2)^2 + (r - r_2)^2 = Ciphertext \quad (6)$$

q_2 and r_2 are second blocks of plaintext. Their values may or may not be equal. It was decrypted by using combinations of equations (6) and (7). The point (q_2, r_2) lies both on circle and hyperboloid. $(p - p_2)$ was calculated by sender using hyperbolic tan as follows. Even if eavesdropper knows about it, the plaintext remains secure. Eavesdropper does not know dimension of hyperboloid namely *l*, *m* and *n*. The number of unknowns which eavesdropper has is more than the number of equations for any block. The same is applicable if the cryptosystem was implemented in the form of a stream cipher. For every *n*th block, eavesdropper has 3n equations, but 3n+3 unknowns. The value of Third block of plaintext was as shown in equation (9).

$$1 = \frac{(p - p_2)^2}{l^2} + \frac{(q - q_2)^2}{m^2} - \frac{(r - r_2)^2}{n^2} \quad (7)$$

$$\tan h = \frac{(p - p_2)}{(r - r_2)} \quad (8)$$

$$q_3 + r_3 = Third\ block\ of\ plaintext \quad (9)$$

$$(q - q_3)^2 + (r - r_3)^2 = Ciphertext\ third\ block \quad (10)$$

It was represented by points (q_3, r_3) which are common on circle and hyperboloid. The values of q_3 and r_3 may or may not be equal. Third block of ciphertext was decrypted by using combinations of equations (10) and (11). *n*th block of plaintext was (q_n, r_n). This point was on surfaces of circle and hyperboloid. Hyperbolic trigonometry was used to decrypt at every block. $(p - p_3)$ was calculated by sender using hyperbolic tan as follows. The value of hyperbolic tan was constant for the complete hyperboloid in equation (12).

$$1 = \frac{(p - p_3)^2}{l^2} + \frac{(q - q_3)^2}{m^2} - \frac{(r - r_3)^2}{n^2} \quad (11)$$

$$\tan h = \frac{(p - p_3)}{(r - r_3)} \quad (12)$$

*n*th block of plaintext was divided into two unequal parts.

$$q_n + r_n = nth\ block\ of\ plaintext \quad (13)$$

*n*th block of ciphertext was calculated as

$$(q - q_n)^2 + (r - r_n)^2 = nth\ Ciphertext\ block \quad (14)$$

$$1 = \frac{(p - p_n)^2}{l^2} + \frac{(q - q_n)^2}{m^2} - \frac{(r - r_n)^2}{n^2} \quad (15)$$

*n*th block of plaintext was encrypted using equation (15). It was decrypted using equations (14) and (15). The value of $(p - p_n)$ was calculated using hyperbolic tan function as in equation (16). The same concepts can be applied in form of stream cipher. It encrypts continuous stream of plaintexts. Stream of plaintext was calculated using equation (17).

$$\tan h = \frac{(p - p_n)}{(q - q_n)} \quad (16)$$

$$q_{stream} + r_{stream} = stream\ of\ plaintext \quad (17)$$

$$(q - q_{stream})^2 + (r - r_{stream})^2 = Stream\ of\ Ciphertext \quad (18)$$

It was decrypted using combinations of equations (18) and (19).

$$1 = \frac{p^2}{l^2} + \frac{(q - q_{stream})^2}{m^2} - \frac{(r - r_{stream})^2}{n^2} \quad (19)$$

III. Advantages of Proposed System

Execution time was calculated till the accuracy of nanoseconds. Throughput was calculated by dividing file size in bytes by execution time in nanoseconds. The proposed system has the following advantages.

- *Higher Randomness in Ciphertext:* Experimental results in terms of video and images have shown that repetitions in blocks of ciphertext are lesser as compared to already existing public key cryptosystems [11]. It is because of higher diffusion and confusion.

- *Reverse Engineering*: The cryptosystem can be implemented on tamper-proof hardware. It is because of an extremely lesser computational complexity. Any kind of attempt of tampering would delete all the contents of the memory.
- *Key Generation, Management, Handling and Distribution*: It was made easier. Any node can advertise it's public and private key without interference from central server. Central Authority can eavesdrop the entire communication. The need for central authority was eliminated. The cryptosystem was designed to be decentralized.
- *Platform Independent Encryption*: The 3D Hyperboloid encryption could work on any kind of hardware. It can be FPGA, ASIC, microcontroller and quantum [12] [13]. The advantages made it easy to implement on software too. The cryptosystem was designed to be easy to code [14]. The worst-case complexity was $O(n^2)$.
- *Compression*: Mathematical properties ensured that ciphertext size was lesser than the plaintext size. Consider Fig. 1. However, in most of the cryptosystems, it was observed that ciphertext was greater than the plaintext.

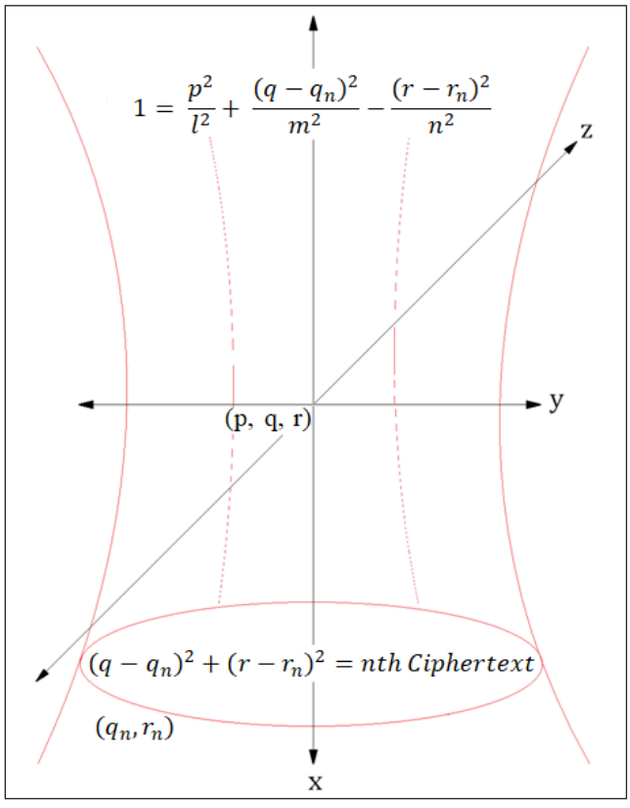

Fig. 1: 3D Hyperbola was used for Decryption. Circle was used for Encryption

Fig. 2: First Image used in Security Analysis

Fig. 3: Second Plaintext Image used in Security Analysis

Fig. 2, Fig. 3 and Fig. 4 shows images used in security analysis. The percentage repetition in cipher text blocks was calculated. The size of the image used in Fig. 2 was 24,375 bytes. The resolution was 448*281. The horizontal resolution was at 266 dpi. Vertical resolution was at 266 dpi. Bit depth was 24. Image used in Fig. 3 was of 1,339,378 bytes. Resolution was at 2688*3772. Horizontal resolution was at 300 dpi. Vertical resolution was at 300 dpi. Bit depth of the image was 24. RGB colour representation was used. File size of the image used in Fig. 4 for experiments was 401,260 bytes.

Resolution was at 628*597. Bit depth was at 24. Table I shows a comparison of most frequently occurring ciphertext blocks with value. It proved that hyperboloid cryptosystem was the most secure.

Fig. 4: Third Plaintext Image used in Security Analysis

Table I: Repetition of Ciphertext Parts with Percentage

Cipher	Most Frequently Occurring Ciphertext Block in Image 2	Percentage of Time it got Repeated in Ciphertext of Image 2	Most Frequently Occurring Ciphertext Block in Image 3	Percentage of Time it got Repeated in Ciphertext of Image 3	Most Frequently Occurring Ciphertext Block in Image 4	Percentage of Time it got Repeated in Ciphertext of Image 4
RSA	01010001	5.43	1010100	6.4534	01111101111	5.4354
AES	00101101	5.434	1100010	5.454	0110001011	6.3543
DES	01110110	6.3543	1100011	4.545	0110101010	7.543543
ECC	00110110	6.354	1110110	5.454	1010111111	5.454
3D Hyperboloid	10001010	4.544	11110101	2.343	100101111	5.14365

Fig. 5: Screen Shot of First Video used in Comparison of Throughput

Fig. 5 shows screenshot of first video used in comparison of throughtput. Comparison was made with already existing public key cryptosystems. The size of the video was 8,914,067 bytes. Resolution of video was 1082*720. It was played at 18 frames per second. Duration of the video was 60 seconds. Video was played at a bit rate of 1185 kbps. Fig. 6 shows second video used in calculation of throughput. Size of the video was 18,409,618 bytes. Frame rate was at 30 frames per second. Resolution of the video was at 1280*720. Duration of the video was 366 seconds. Bit rate was at 397 kbps. Fig. 7 shows screenshot of third video used in throughput comparison. It was of 51,299,343 bytes. Duration of the video was 221 seconds. Resolution was at 1280*676. Bit rate was at 1853 kbps. It was played at 23 frames per second.

Fig. 8 shows fourth video used in calculation of throughput. File size was at 3,200,087 bytes. Video was of 52 seconds. Resolution of the video was at 1280*720. Bit rate was at 479 kbps. Video was played at 29 frames per second. Fig. 9 is screenshot of fifth video used in experiments. Size of the video was 20,819,208 bytes. Duration of the video was 232 seconds. Resolution of the video was 1280*720. Total bit rate was 713 kbps. It was played at 29 frames per second. Table II shows comparison in bytes per second for various cryptosystems. 3D hyperboloid cryptosystem was the fastest.

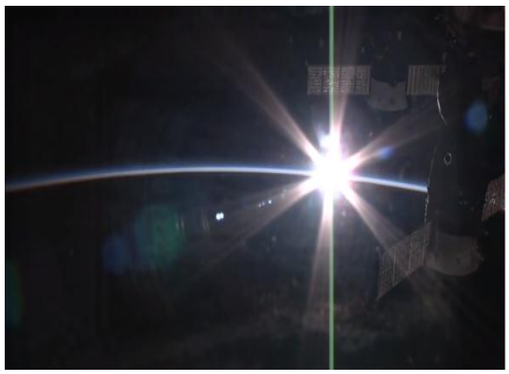

Fig. 6: Screen Shot of Second Video used in Comparison of Throughput

Fig. 7: Screen Shot of Third Video used in Comparison of Throughput

Table II: Comparison of Throughput in Bytes Per Second

Cipher	Throughput for Video in Fig. 5	Throughput for Video in Fig. 6	Throughput for Video in Fig. 7	Throughput for Video in Fig. 8	Throughput for Video in Fig. 9
RSA	1,342,453	3,342,434	3,967,444	3,856,453	3,575,264
AES	5,454,235	5,454,656	5,954,435	5,894,546	5,868,545
DES	6,465,436	5,455,454	5,856,756	5,788,673	5,626,236
ECC	4,545,434	3,434,345	4,545,786	4,567,364	4,456,346
3D Hyperboloid	6,554,545	5,553,449	5,995,457	5,967,363	5,965,653

Fig. 8: Screen Shot of Fourth Video used in Comparison of Throughput

Fig. 9: Screen Shot of Fifth Video used in Comparison of Throughput

IV. Conclusion

In future implementation, digital signature and digital envelope can be added. Experimental results have demonstrated that the proposed cryptosystem is better than both symmetric and asymmetric key cryptosystems. More cryptosystems can be added in future work for comparison of speed and security. Symmetric key cryptography was fast but insecure. Asymmetric key cryptography was slow but was vulnerable to side-channel attacks. The novel cryptosystem was designed to solve these problems. Principles of trigonometry were used to ensure that plaintext was larger than ciphertext. It was the fastest public key cryptosystem. Testing was done on different types of file extensions. It was easy to change keys due to lesser complexity.

References

[1] R. Hodgson, "Solving the security challenges of IoT with public key cryptography," *Network Security*, vol. 2019, no. 1, pp. 17-19, 2019.

[2] R. Thiyagarajan, and B. M. Priya, "An enhancement of EAACK using P2P ACK and RSA public key cryptography," *Measurement*, vol. 136, pp. 116-121, Mar. 2019.

[3] U. Vazirani, and T. Vidick, "Fully device independent quantum key distribution," *Communications of the ACM*, vol. 62, no. 4, pp. 133-133, 2019.

[4] C. Majenz, C. Schaffner, and J. van Wier, "Non-malleability for quantum public-key encryption," *Quantum Physics*, May 2019.

[5] X.-L. Pang, L.-F. Qiao, K. Sun, Y. Liu, A.-Lin Yang, and X.-M. Jin, "Experimental quantum-enhanced cryptographic remote control," *Scientific Reports*, vol. 9, 2019, Art. no. 5809.

[6] P. S. Goswami, and T. Chakraborty, "Design of a quantum one-way trapdoor function," *Emerging Technology in Modelling and Graphics*, Springer, pp. 547-555, Jul. 2019.

[7] P. Xu, S. He, W. Wang, W. Susilo, and H. Jin, "Lightweight searchable public-key encryption for cloud-assisted wireless sensor networks," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 8, 2018.

[8] D. Basin, C. Cremers, T. H.-J. Kim, A. Perrig, R. Sasse, and P. Szalachowski, "Design, analysis and implementation of ARPKI: An attack-resilient public-key infrastructure," *IEEE Transactions on Dependable and Secure Computing*, vol. 15, no. 3, 2018.

[9] U. Banerjee, A. Pathak, and A. P. Chandrakasan, "An energy-efficient configurable lattice cryptography processor for the quantum-secure internet of things," *IEEE Int. Solid-State Circuits Conf. (ISSCC),* 2019.

[10] Z. Jing, C. Gu, and P. Shi, "Cryptanalysis of a public key cryptosystem based on data complexity under quantum environment," *Int. Conf. Secur. and Privacy New Comput. Environ. (SPNCE),* 2019, pp 411-420.

[11] S. E. Smart, D. I. Schuster, and D. A. Mazziotti, "Experimental data from a quantum computer verifies the generalized Pauli exclusion principle," *Communications Physics*, vol. 2, 2019, Art. no. 11.

[12] X. Xin, Z. Wang, Q. H. Q. Yang, and F. Li, "New public-key quantum signature scheme with quantum one-way function," *International Journal of Theoretical Physics*, pp. 1-13, Jul. 2019.

[13] V. Gheorghiu, and M. Mosca, "Benchmarking the quantum cryptanalysis of symmetric, public-key and hash-based cryptographic schemes," *Quantum Physics*, 2019.

[14] X.-Q. Cai, T.-Y. Wang, C.-Y. Wei, and F. Gao, "Cryptanalysis of multiparty quantum digital signatures," *Quantum Information Processing,* vol. 18, Aug. 2019, Art. no. 252.