A Post-Quantum Encryption Approach (PQEA)

Osvaldo Skliar* Sherry Gapper† Ricardo E. Monge‡

December 15, 2024

Abstract

A Post-Quantum Encryption Approach (PQEA) is presented. Its main advantages are 1) its high level of efficiency and effectiveness, and 2) the simplicity of its development and use. In messages encrypted with the PQEA there are no visible patterns or regularities which could give any idea of the corresponding original messages.

Keywords: post-quantum cryptography; quantum-resistant cryptography; random number generator; random binary sequences
Mathematics Subject Classification 2020: 11T71, 68P25, 94A60

1 Introduction

Experts in computer security usually consider RSA (Ron *Rivest*, Adi *Shamir*, Leonard *Adleman*) to be the strongest of the cryptographic methods currently available. It is based on the notable difficulty to factor the product of two very large prime numbers into its prime numbers [1, 2]. However, many of these experts believe that the development of quantum computing will make it possible, by using computer programs based on Shor's algorithm, to decipher messages encrypted with the RSA cryptographic system [3, 4]. Not all authors share that opinion [5, 6].

Regardless of that controversy, it is useful to have a quantum-resistant cryptographic approach such as that presented in this article. The Post-Quantum Encryption Approach (PQEA) can be considered as a significant improvement of [7], itself of very high quality.

Messages encrypted by the PQEA have no regularities or patterns which could offer any orientation about the original messages. This result is obtained by a) certain permutations of the bytes and of the bits comprising the messages

*Universidad Nacional, Costa Rica. E-mail: osvaldoskliar@gmail.com. https://orcid.org/0000-0002-8321-3858.
†Universidad Nacional, Costa Rica. E-mail: sherry.gapper.morrow@una.ac.cr. https://orcid.org/0000-0003-4920-6977.
‡Universidad CENFOTEC, Costa Rica. E-mail: rmonge@ucenfotec.ac.cr. https://orcid.org/0000-0002-4321-5410.

being encrypted, and b) the use of diverse random binary sequences. To carry out these operations and generate these sequences, use is made of two random number generators: the Hybrid Random Number Generator (HRNG) [8] and the Mathematical Random Number Generator (MRNG) [9]. [10]. As an introduction to cryptography topics, the reader may consult, for example, [10], [11] and [12].

2 Description of the message encryption process using the PQEA

2.1 Determining the number of bytes in the sequence of bytes comprising the message to be encrypted and how to proceed according to that number

The number of bytes in the sequence of bytes comprising the message to be encrypted is counted. That number is denominated N. If $N \geq 100$ (that is, if N is greater than 100 or equal to 100), step 2.1 in the process described is considered terminated. If, on the other hand, $N < 100$ (that is, if N is less than 100), another sequence of $(100 - N)$ bytes is added, or concatenated, to the sequence of bytes considered. This sequence consists of $(100 - N)$ repetitions of the only byte corresponding to the whitespace character (00100000). Thus, for example, if the message to be encrypted is composed of a sequence of 87 bytes, another sequence of $(100 - 87)$ bytes (that is, 13 bytes) is concatenated to the former sequence of (100-87) bytes. The latter sequence is composed of 13 repetitions of the only byte corresponding to the whitespace character.

Upon concluding the decryption process of the encrypted message, the eventual presence of a repetitive sequence of bytes, each of which is the byte corresponding to the whitespace character, does not interfere with the comprehension of the decrypted message because it has the same appearance as that of the original message that underwent the encryption process.

2.2 Random selection of a permutation from the 256! permutations possible of the 256 existing bytes

Recall that each byte is composed of a sequence of 8 bits, each of which may be a 0 (zero) or a 1 (one). Thus there are 2^8, or 256, different bytes.

Those bytes can be ordered by interpreting each of them as a number expressed in the binary number system (that is, in base 2). Therefore, if the numbers corresponding to those bytes are considered in increasing order, the first of these numbers is 00000000 and the last of them is 11111111.

When carrying out a permutation of these 256 bytes, for example, the first of them or byte 1 (00000000) could be replaced by the seventeenth of them or byte 17 (00010000); byte 2 (00000001) could be replaced by byte 109 (01101100), and so on, until byte 256 (11111111) is replaced by byte 95 (01011110). Note that,

given that 00000000 was denominated byte 1, byte n, for $n = 1, 2, 3, \ldots, 256$, will correspond to the sequence of bits which, when interpreted as a number expressed in base 2, is $n - 1$.

2.3 Replacing each of the bytes in the sequence of bytes to be encrypted (that resulting from the process specified in 2.1) with the bytes resulting from the permutation specified in 2.2

If, according to the permutation specified in 2.2, byte 1 is replaced by byte 17, byte 2 is replaced by byte 109, and so on successively until byte 256 is replaced by byte 95, then in the sequence of bytes to be encrypted (that resulting from the process specified in 2.1), bytes $1, 2, \ldots$ and 256, wherever they are present, are replaced by bytes, $17, 109, \ldots$ and 95, respectively. In other words, the random permutation obtained in 2.2 is used to make the pertinent replacements in the sequence of bytes to be encrypted (that resulting from the process specified in 2.1).

Note that carrying out the decryption process described above is very simple because it suffices to use the "inverse permutation": byte 17 is replaced by byte 1, byte 109 is replaced by byte 2, and so on until byte 95 is replaced by byte 256.

2.4 Random permutation altering the order of the bytes in the message being encrypted, that resulting from the process specified in 2.3

According to 2.1, the minimum number of bytes in a message being encrypted is equal to 100. Of course it can be greater than 100. This sequence of bytes being encrypted (that resulting from the process specified in 2.3) undergoes a permutation of the order in which those bytes appear.

Using the PQEA, the authors proceeded as follows to carry out this permutation.

A "window" composed of or covering 100 bytes is used. That "window" is initially placed so that it covers the first 100 bytes in the sequence of bytes being encrypted. Those 100 bytes undergo a permutation of their order in that "window." That permutation is selected randomly among the 100! possible permutations.

To indicate the positions that the bytes occupy initially in that "window" covering 100 bytes, the following notation is used: (byte 1)*, (byte 2)*, ..., (byte 100)* are the bytes occupying, respectively, the first position, the second position, ..., and the hundredth position in the "window" before carrying out the random permutation of their positions that are being considered. This notation is adopted to prevent confusion due to the notation used above in 2.2, according to which byte 1, byte 2, ..., and byte 256 referred to the numerical value, expressed in base 2, of each of those bytes.

Suppose that, given the random permutation selected, (byte 1)* takes the place, or position, of (byte 57)*, (byte 2)* takes the position of (byte 100)*, etc., and finally (byte 100)* takes the position of (byte 39)*.

If the sequence of bytes to be encrypted has only 100 bytes, the process described in 2.4 would be considered terminated.

If that sequence of bytes to be encrypted has more than 100 bytes, the entire sequence is "swept" with that "window," completing a "pass" or displacement, of one byte on each occasion. In each new position of the "window" the same permutation as that specified above is carried out: The byte in the first position of that "window" (byte 1)* takes the place, or the position, of (byte 57)*, (byte 2)* takes the place of (byte 100)*, etc., and finally (byte 100)* takes the place of (byte 39)*. The last position occupied by the "window" considered, and in which the permutation specified occurs, is of course, that which covers the last 100 bytes in the encryption process.

Suppose that the number of bytes in that sequence of bytes being encrypted is equal to 2,700. As accepted, the "window" used to "sweep" that sequence covers 100 bytes. In this case, the number of different positions that this "window' will occupy is equal to $(2,700 - 100 + 1)$, that is, 2,601; and the number of displacements of that "window" from its initial position, in which it covers the first 100 bytes of that sequence of 2,700 bytes, until it reaches the last position in which it covers the last 100 bytes in that sequence, is equal to 2,600.

In general, if the number of bytes covering the "window" is N_W, and the number of bytes in the sequence being encrypted is N_B, the number of different positions that the "window" will occupy (in each of which the specified random permutation of positions of the bytes is produced) is $(N_B - N_W + 1)$ and the number of displacements of the "window" (each of which is one byte) is $(N_B - N_W)$.

The decryption process corresponding to the encryption specified in 2.4 (described above) is obvious: A "sweep" is carried out with the "window" in the inverse direction (that is, as of its final position until it reaches its initial position) and in each of these positions a permutation "inverse" to that of the encryption is carried out. In other words, in each of the positions in the "window" during the "inverse sweep", (byte 57)* takes the place of (byte 1)*, (byte 100)* takes the place of (byte 2)*, etc., until (byte 39)* takes the place of (byte 100)*.

2.5 Random permutation altering the order of the bits in the message being encrypted (that resulting from the process specified in 2.4)

The characterization of this random permutation carried out after that of 2.4 is very direct because it operates like the latter with the following differences: a) instead of using a "window" which covers 100 bytes, it uses a "window" which covers 100 bits with which it "sweeps" the entire sequence of bits comprising the message being encrypted; b) a new random permutation is used to alter the

places of the 100 bits covered by the "window" in each of the positions occupied by that window during the "sweeping" process; and c) each displacement of the "window" is one bit and not one byte (as in 2.4).

In general, if W_b is the number of bits covered by the "window" used and N_b is the number of bits in the sequence of bits comprising the message being encrypted, the number of different positions taken by the "window" is equal to $(N_b - W_b + 1)$; and the number of displacements of that "window" from its initial position (covering the first 100 bits in the sequence of bits being encrypted) up to its final position (covering the last 100 bits in the sequence) is $(N_b - W_b)$. In particular, if N_B is 2,700, N_b is 8. $N_B = 21,600$ and the number of positions taken by the "window" being "swept" is $(21,600 - 100 + 1) = 21,501$. In this case, the number of displacements of the "window" during the "sweeping" process is 21,500.

The decryption process corresponding to the the encryption process specified in 2.5 is obvious: An inverse "sweep" is carried out with the "window" of the sequence of bits considered, and in each position of the "window" in that "inverse sweep," the "inverse permutation" of the bits is applied to that done during the encryption process.

2.6 Using random binary sequences in the final step in encrypting a given message

The output of the encryption process specified in 2.5 is a sequence of the bits that will be called G_0.

Various random binary sequences generated using the HRNG of the MRNG have been prepared for this final step in the message encryption process. In the preparation of the cryptographic systems described, 3 of these sequences have been used in each of these systems. Each has a minimum number of 800,000 bits. The first, second and third of these sequences, according to the order in which they are used, are denominated S_1, S_2 and S_3, respectively.

G_0 is compared bit to bit to S_1, and the operation is carried out according to two possibilities (A and B) to be specified below. A random choice is made as to which of these two possibilities will be used. The probability that A will be chosen is the same as the probability that B will be chosen. In other words, the probability of selecting each of those two possibilities is $\frac{1}{2}$. How each possibility operates will be specified below.

If A is selected, for two bits that are the same, in the comparison mentioned above, the output is a 1, and for two different bits, the output is a 0. In other words, if A is selected, both 00 and 11 generate a 1 as output; and both 01 and 10 generate a 0 as output. On the other hand, if B is selected, for two bits that are the same, the output is a 0, and for two different bits, the output is a 1. In other words, if B is selected, both 00 and 11 generate a 0 as output; and both 01 and 10 generate a 1 as output.

G_0 is compared bit to bit with S_1 and operating according to whether A or B has been selected, a binary sequence G_1 is generated as output.

G_1 is compared bit to bit with S_2 and operating according to whether A or B has been selected, a binary sequence G_2 is generated as output.

G_2 is compared bit to bit with S_3 and operating according to whether A or B has been selected, a binary sequence G_3 is generated as output.

G_3 is the entirely encrypted message.

In figures 1, 2 and 3, the above processes are illustrated for the first 8 bits of the different binary sequences considered.

$01100011\ldots$: First 8 bits of G_0
$10110100\ldots$: First 8 bits of S_1
$00101000\ldots$: First 8 bits of G_1

Figure 1: In this case possibility A was randomly selected. As output, a 1 corresponds to both 00 and 11, and a 0 corresponds to both 01 and 10.

$00101000\ldots$: First 8 bits of G_1
$01001011\ldots$: First 8 bits of S_2
$01100011\ldots$: First 8 bits of G_2

Figure 2: In this case possibility B was randomly selected. As output, a 0 corresponds to both 00 and 11, and a 1 corresponds to both 01 and 10.

$01100011\ldots$: First 8 bits of G_2
$11001100\ldots$: First 8 bits of S_3
$10101111\ldots$: First 8 bits of G_3

Figure 3: In this case possibility B was randomly selected. As output, a 0 corresponds to both 00 and 11, and a 1 corresponds to both 01 and 10. G_3 is the entirely encrypted message.

Suppose that G_0 has more than 800,000 bits, the number of bits in each of the random binary sequences S_1, S_2 and S_3 already prepared. In this case one begins to reuse these sequences starting with their initial bits. In other words, the bit that occupies place 800,001 in the sequence of bits being encrypted will be compared to the first bit in the respective random binary sequence already prepared; the bit that occupies place 800,002 in the sequence of bits being encrypted will be compared to the second bit in the corresponding random binary sequence, and so on successively.

The diagram in figure 4 illustrates how this works so that the bits needed from S_1, S_2 and S_3 will not be "used up."

It is simple to "revert" the encryption process which uses S_1, S_2 and S_3. For that purpose (that is, to carry out the decryption process corresponding to the encryption process specified in 2.6), one does as follows: 1) Given G_3 and S_3 and the operational possibility chosen, G_2 is computed; 2) given G_2 and S_2

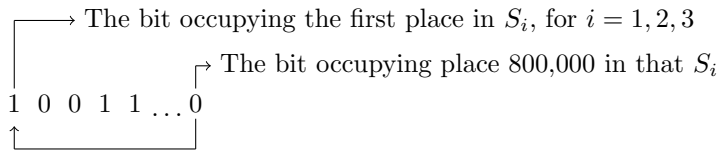


Figure 4: When encrypting a message, if it becomes necessary (something that occurs when G_0 has more than 800,000 bits), each S_i, for $i = 1, 2, 3$ is reused.

and the operational possibility chosen, G_1 is computed; and 3) given G_1 and S_1 and the operational possibility chosen, G_0 is computed. Recall that G_0 is the output of the encryption process specified in 2.5.

3 Examples of encryption using a PQEA-type system

3.1 Examples using a PQEA-type system to encrypt and decrypt two short texts

First, the abstract from [9] is reproduced here; and under it, the respective encryption. Following that, the abstract from [8] is transcribed; and under it, the corresponding encryption is reproduced. Spaces have been added here in both encrypted texts only to ease readability here. Of course, those spaces do not appear in the entirely encrypted message that the sender transmits to its recipient(s). exchanged between the sender and its recipient(s).

Text 1 to be encrypted:

A novel Mathematical Random Number Generator (MRNG) is presented here. In this case, "mathematical" refers to the fact that to construct that generator it is not necessary to resort to a physical phenomenon, such as the thermal noise of an electronic device, but rather to a mathematical procedure. The MRNG generates binary strings – in principle, as long as desired – which may be considered genuinely random in the sense that they pass the statistical tests currently accepted to evaluate the randomness of those strings. From those strings, the MRNG also generates random numbers expressed in base 10. An MRNG has been installed as a facility on this web page. This generator may be used for applications in tasks in: a) computational simulation of probabilistic-type systems, and b) the random selection of samples of different populations. Users interested in applications in cryptography can build another MRNG, but they would have to withhold information – specified in section 5 – from people who are not authorized to decode messages encrypted using that resource.

Encryption 1:

10011001 00110100 11001011 01000110 00111100 00110111 11111000 10000101 01100100 00101011 00100110 11111100
11000000 01000100 00101011 00100110 11111101 11000110 01001000 00100110 01110010 11000110 11000100 01000111
00101110 00111101 11111001 10000101 01100111 00111111 00111111 11110110 11000000 01011011 01101010 00010101
11110001 11001011 01001100 00111000 00110011 11100000 11001010 01011011 01101010 01111010 11011001 11110111
01100111 00001101 01111011 10110100 11001100 01011010 01101010 00100010 11100110 11000000 01011010 00101111
00111100 11100000 11000000 01001101 01101010 00111010 11110001 11010111 01001100 01100100 01110010 11011101
11001011 00001001 00111110 00111010 11111101 11010110 00001001 00101001 00110011 11100111 11000000 00000101
01101010 01110000 11111001 11000100 01011101 00100010 00110111 11111001 11000100 01011101 00100011 00110001
11110101 11001001 00001011 01101010 00100000 11110001 11000011 01001100 00111000 00100001 10110100 11010001
01000110 01101010 00100110 11111100 11000000 00001001 00101100 00110011 11110111 11010001 00001001 00111110
00111010 11110101 11010001 00001001 00111110 00111101 10110100 11000110 01000110 00100100 00100001 11100000
11010111 01011100 00101001 00100110 10110100 11010001 01000001 00101011 00100110 10110100 11000010 01001100
00100100 00110111 11100110 11000100 01011101 00100101 00100000 10110100 11001100 01011101 01101010 00111011
11100111 10000101 01000111 00100101 00100110 10110100 11001011 01001100 00101001 00110111 11100111 11010110
01001000 00111000 00101011 10110100 11010001 01000110 01101010 00100000 11110011 01010110 00111000 00111000
00100110 10110100 11010001 01000110 01101010 00110011 10110100 11010101 01000001 00110011 00100001 11111101
11000110 01001000 00100110 01110010 11100100 11001101 01001100 00100100 00111101 11111001 11000000 01000111
00100101 00111100 10111000 10000101 01011010 00111111 00110001 11111100 10000101 01001000 00111001 01110010
11100000 11001101 01001100 01101010 00100110 11111100 11000000 01011011 00100111 00110011 11111000 10000101
01000111 00100101 00111011 11100111 11000000 00001001 00100101 00110100 10110100 11000100 01000111 01101010
00110111 11111000 11000000 01001010 00111110 00100000 11111011 11001011 01000000 00101001 01110010 11110000
11000000 01011111 00100011 00110001 11110001 10001001 00001001 00101000 00100111 11100000 10000101 01011011
00101011 00100110 11111100 11000000 01011011 01101010 00100110 11111011 01001000 00101001 01101010 00111111
11110101 11010001 01000001 00101111 00111111 11110101 11010001 01000000 00101001 00110011 11111000 10000101
01011001 00111000 00111101 11110111 11000000 01001101 00111111 00100000 11110001 10001011 00001001 00111110
00111010 11110001 10000101 01100100 00011000 00011100 11010011 10000101 01001110 00101111 00111100 11110001
11010111 01001000 00111110 00110111 11100111 10000101 01001011 00100011 00111100 11110101 11010111 01010000
01101010 00100001 11100000 11010111 01000000 00100100 00110101 11100111 10000101 00000100 01101010 00111011
11111010 10000101 01011001 00111000 00111011 11111010 11000110 01000000 00111010 00111110 11110001 10001001
00001001 00101011 00100001 10110100 11001001 01000110 00100100 00110101 10110100 11000100 01011010 01101010
00110110 11110001 11010110 01000000 00111000 00110111 11110000 10000101 00000100 01101010 00100101 11111100
11001100 01001010 00100010 01110010 11111001 11000100 01010000 01101010 00110000 11110001 10000101 01001010
00100101 00111100 11100111 11001100 01001101 00101111 00100000 11110001 11000001 00001001 00101101 00110111
11111010 11010000 01000000 00100100 00110111 11111000 11011100 00001001 00111100 00111010 11110001 11000001
01000110 00100111 01110010 11111101 11001011 00001001 00111110 00111010 11110101 11010001 00001001 00111110
00111100 11100111 11000000 00001001 00111110 00111010 11110101 11010001 00001001 00111110 00111010 11110001
11011100 00001001 00111010 00110011 11100111 11010110 00001001 00111110 00111010 11110001 10000101 01011010
00111110 00110011 11100000 11001100 01011010 00111110 00111011 11110111 11000100 01000101 01101010 00100110
11110001 11010110 01011101 00111001 01110010 11110111 11010000 01011011 00111000 00110111 11111010 11010001
01000101 00110011 01110010 11110101 11000110 01001010 00101111 00100010 11100000 11000000 01001101 01101010
00100110 11111011 10000101 01001100 00111100 00110011 11111000 11010000 01001000 00111100 00110111 10110100
11010001 01000001 00101111 01110010 11100110 11000100 01000111 00101110 00111101 11111001 11001011 01001100
00111001 00100001 10110100 11001010 01001111 01101010 00100110 11111100 11001010 01011010 00101111 01110010
11100111 11010001 01011011 00100011 00111100 11110011 11010110 00000111 01101010 00010100 11100110 11001010
01000100 01101010 00100110 11111100 11001010 01011010 00101111 01110010 11100111 11010001 01011011 00100011
00111100 11110011 11010110 00000101 01101010 00100110 11111100 11000000 00001001 00000111 00000000 11011010
11100010 00001001 00101011 00111110 11100111 11001010 00001001 00101101 00110111 11111010 11000000 01011011
00101011 00100110 11110001 11010110 00001001 00111000 00110011 11111010 11000001 01000110 00100111 01110010
11111010 11010000 01000100 00101000 00110111 11100110 11010110 00001001 00101111 00101010 11100100 11010111
01001100 00111001 00100001 11110001 11000001 00001001 00100011 00111100 10110100 11000111 01001000 00111001
00110111 10110100 10010100 00011001 01100100 01110010 11010101 11001011 00001001 00000111 00000000 11011010
11100010 00001001 00100010 00110011 11100111 10000101 01001011 00101111 00110111 11111010 10000101 01000000
00100100 00100001 11100000 11000100 01000101 00100110 00110111 11110000 10000101 01001000 00111001 01110010
11110101 10000101 01001111 00101011 00110001 11111101 11001001 01000000 00111110 00101011 10110100 11001010
01000111 01101010 00100110 11111100 11001100 01011010 01101010 00100101 11110001 11000111 00001001 00111010
00110011 11110011 11000000 00000111 01101010 00000110 11111100 11001100 01011010 01101010 00110101 11110001
11001011 01001100 00111000 00110011 11100000 11001010 01011011 01101010 00111111 11110101 11011100 00001001
00101000 00110111 10110100 11010000 01011010 00101111 00110110 10110100 11000011 01000110 00111000 01110010
11110101 11010101 01011001 00100110 00111011 11110111 11000100 01011101 00100011 00111101 11111010 11010110
00001001 00100011 00111100 10110100 11010001 01001000 00111001 00111001 11100111 10000101 01000000 00100100
01101000 10110100 11000100 00000000 01101010 00110001 11111011 11001000 01011001 00111111 00100110 11110101
11010001 01000000 00100101 00111100 11110101 11001001 00001001 00111001 00111011 11111001 11010000 01000101
00101011 00100110 11111101 11001010 01000111 01101010 00111101 11110010 10000101 01011001 00111000 00111101
11110110 11000100 01001011 00100011 00111110 11111101 11010110 01011101 00100011 00110001 10111001 11010001
01010000 00111010 00110111 10110100 11010110 01010000 00111001 00100110 11110001 11000110 01011010 01100110
01110010 11110101 11001011 01001101 01101010 00110000 10111101 10000101 01011101 00100010 00110111 10110100
11010111 01001000 00100100 00110110 11111011 11001000 00001001 00111001 00110111 11111000 11000000 01001010
00111110 00111011 11111011 11001011 00001001 00100101 00110100 10110100 11010110 01001000 00100111 00100010
11111000 11000000 01011010 01101010 00111101 11110010 10000101 01001101 00100011 00110100 11110010 11000000
01011011 00101111 00111100 11100000 10000101 01011001 00100101 00100010 11100001 11001001 01001000 00111110
00111011 11111011 11001011 01011010 01100100 01110010 11000001 11010110 01001100 00111000 00100001 10110100
11001100 01000111 00111110 00110111 11100110 11000000 01011010 00101111 00110110 10110100 11000011 01000110
00100100 01110010 11110101 11010101 01011001 00100110 00111011 11110111 11000100 01011101 00100011 00111101
11111010 11010110 00001001 00110011 00111100 10110100 11000110 01011011 00110011 00100010 11100000 11001010
01001110 00111000 00110011 11100100 11001101 01010000 01101010 00110001 11110101 11001011 00001001 00101000
00100111 11111101 11001001 01001101 01101010 00110011 11111010 11001010 01011101 00100010 00110111 11100110
10000101 01100100 00011000 00011100 11010011 10001001 00001001 00101000 00100111 11100000 10000101 01011101
00100010 00110111 11101101 10000101 01011110 00100101 00100111 11111000 11000001 00001001 00100010 00110011
11100010 11000000 00001001 00111110 00111101 10110100 11010010 01000000 00111110 00110111 11110000 10000101 01000000
01000101 00101110 01110010 11111101 11001011 01001111 00100101 00100000 11111001 11000100 01011101 00100011
00111101 11111010 10000101 00000100 01101010 00100001 11100100 11000000 01001010 00100011 00110100 11111101
11000000 01001101 01101010 00111011 11111010 10000101 01011010 00101111 00110001 11100000 11001100 01000110
00100100 01110010 10100001 10000101 00000100 01101010 00110100 11100110 11001010 01000100 01101010 00100010
11110001 11001010 01011001 00100110 00110111 10110100 11010010 01000001 00100101 01110010 11110101 11010111
01001100 01101010 00111100 11111011 11010001 00001001 00101011 00100111 11100000 11001101 01000110 00111000
00111011 11101110 11000000 01001101 01101010 00100110 11111011 10000101 01001101 00101111 00110001 11111011
11000001 01001100 01101010 00111111 11110001 11010110 01011010 00101011 00110101 11110001 11010110 00001001
00101111 00111100 11110111 11010111 01010000 00111010 00100110 11110001 11000001 00001001 00111111 00100001
11111101 11001011 01001110 01101010 00100110 11111100 11000100 01011101 01101010 00100000 11110001 11010110
01000110 00111111 00100000 11110111 11000000 00000111

Text 1 as decrypted:

A novel Mathematical Random Number Generator (MRNG) is presented here. In this case, "mathematical" refers to the fact that to construct that generator it is not necessary to resort to a physical phenomenon, such as the thermal noise of an electronic device, but rather to a mathematical procedure. The MRNG generates binary strings – in principle, as long as desired – which may be considered genuinely random in the sense that they pass the statistical tests currently accepted to evaluate the randomness of those strings. From those strings, the MRNG also generates random numbers expressed in base 10. An MRNG has been installed as a facility on this web page. This generator may be used for applications in tasks in: a) computational simulation of probabilistic-type systems, and b) the random selection of samples of different populations. Users interested in applications in cryptography can build another MRNG, but they would have to withhold information – specified in section 5 – from people who are not authorized to decode messages encrypted using that resource.

Text 2 to be encrypted:

The purpose of this paper is to present a novel Hybrid Random Number Generator (HRNG). Here hybrid refers to the fact that to construct this generator it is necessary to use 1) physical components texts and a physical process, and 2) a mathematical procedure. This HRNG makes it possible to generate genuine random numbers which may be used both for computer simulation of probabilistic systems and in the field of cryptography. The results of a comparative study of the binary strings generated by this HRNG and of those generated by two highly used implementations of a congruential algorithm designed to generate pseudorandom numbers are given here. One of the latter is the implementation incorporated into the Java 2 platform (version 1.6), and the other is the implementation incorporated into the runtime library of Microsofts Visual C++ 2008 compiler.

Encryption 2:

10001100 01111100 11000000 00001001 00111010 00100111 11100110 11010101 01000110 00111001 00110111 10110100
11001010 01001111 01101010 00100110 11111100 11001100 01011010 01101010 00100010 11110101 11010101 01001100
00111000 01110010 11111101 11010110 00001001 00111110 00111101 10110100 11010101 01011011 00101111 00100001
11110001 11001011 01011101 01101010 00110011 10110100 11001011 01000110 00111100 00110111 11111000 10000101
01100001 00110011 00110000 11100110 11001100 01001101 01101010 00000000 11110101 11001011 01001101 00100101
00111111 10110100 11101011 01011100 00100111 00110000 11110001 11010111 00001001 00001101 00110111 11111010
11000000 01011011 00101011 00100110 11111011 11010111 00100100 01000000 01110010 10111100 11101101 01111011
00000100 00010101 10111101 10001011 00001001 00000010 00110111 11100110 11000000 00001001 00100010 00101011
11110110 11010111 01000000 00101110 01110010 11100110 11000000 01001111 00101111 00100000 11100111 10000101
01011101 00100101 01110010 11100000 11001101 01001100 01101010 00110100 11110101 11000110 01011101 01101010
00100110 11111100 11000100 01011101 01101010 00100110 11111011 10000101 01001010 00100101 00111100 11100111
11010001 01011011 00111111 00110001 11100000 10000101 01011101 00100010 00111011 11100111 10000101 01001110
00101111 00111100 11110001 11010111 01001000 00111110 00111101 11100110 10000101 01000000 00111110 01110010
11111101 11010110 00001001 00100100 00110111 11110111 11000000 01011010 00111001 00110011 11100110 11011100
00100100 01000000 01110010 11100000 11001010 00001001 00111111 00100001 11110001 10000101 00011000 01100011
01110010 11100100 11001101 01010000 00111001 00111011 11110111 11000100 01000101 01101010 00110001 11111011
11001000 01011001 00100101 00111100 11110001 11001011 01011101 00111001 01110010 11100000 11000000 01010001
00111110 00100001 10110100 11000100 01000111 00101110 01110010 11110101 10000101 01011001 00100010 00101011
11100111 11001100 01001010 00101011 00111110 10110100 11010101 01011011 00100101 00110001 11110001 11010110
01011010 01100110 01110010 11110101 11001011 01001101 01101010 01100000 10111101 10000101 01001000 01101010
00111111 11110101 11010001 01000001 00101111 00111111 11110101 11010001 01000000 00101001 00110011 11111000
10101000 00100011 01101010 00100010 11100110 11001010 01001010 00101111 00110110 11100001 11010111 01001100
01100100 01110010 11000000 11001101 01000000 00111001 01110010 11011100 11110111 01100111 00001101 01110010
11111001 11000100 01000010 00101111 00100001 10110100 11001100 01011101 01101010 00100010 11111011 11010110
01011010 00100011 00110000 11111000 11000000 00001001 00111110 00111101 10110100 11000010 01001100 00100100
00110111 11100110 11000100 01011101 00101111 01110010 11110011 11000000 01000111 00111111 00111011 11111010
11000000 00001001 00111000 00110011 11111010 11000001 01000110 00100111 01110010 11111010 11010000 01000100
00101000 00110111 11100110 11010110 00001001 00111101 00111010 11111101 11000110 01000001 01000111 01011000

10110100 11001000 01001000 00110011 01110010 11110110 11000000 00001001 00111111 00100001 11110001 11000001
00001001 00101000 00111101 11100000 11001101 00001001 00101100 00111101 11100110 10000101 01001010 00100101
00111111 11100100 11010000 01011101 00101111 00100000 10110100 11010110 01000000 00100111 00100111 11111000
11000100 01011101 00100011 00111101 11111010 10000101 01000110 00101100 01110010 11100100 11010111 01000110
00101000 00110011 11110110 11001100 01000101 00100011 00100001 11100000 11001100 01001010 01101010 00100001
11101101 11010110 01011101 00101111 00111111 11100111 10000101 01001000 00100100 00110110 10110100 11001100
01000111 01101010 00100110 11111100 11000000 00001001 00101111 00111110 11110000 10000101 01000110 00101100
01011111 10011110 10000101 01001010 00111000 00101011 11100100 11010001 01000110 00101101 00100000 11110101
11010101 01000001 00110011 01111100 10110100 11110001 01000001 00101111 01110010 11100110 11000000 01011010
00111111 00111110 11100000 11010110 00001001 00100101 00110100 10110100 11000100 00001001 00101001 00111101
11111001 11010101 01001000 00111000 00110011 11100000 11001100 01011111 00101111 01110010 11100111 11010001
01011100 00101110 00101011 10110100 11001010 01001111 01101010 00100110 11111100 11000000 00001001 00101000
00111011 11111010 11000100 01011011 00110011 01110010 11100111 11010001 01011011 00100011 00111100 11110011
11010110 00001001 00101101 00110111 11111010 11000000 01011011 00101011 00100110 11110001 11000001 00001001
00101000 00101011 10110100 11010001 01000001 00100011 00100001 10011001 10101111 00001001 00000010 00000000
11011010 11100010 00001001 00101011 00111100 11110000 10000101 01000110 00101100 01110010 11100000 11001101
01000110 00111001 00110111 10110100 11000010 01001100 00100100 00110111 11100110 11000100 01011101 00101111
00110110 10110100 11000111 01010000 01101010 00100110 11100011 11001010 00001001 00100010 00111011 11110011
11001101 01000101 00110011 01110010 11100001 11010110 01001100 00101110 01110010 11111101 11001000 01011001
00100110 00110111 11111001 11000000 01000111 00111110 00110011 11100000 11001100 01000110 00100100 00100001
10110100 11001010 01001111 01101010 00110011 10110100 11000110 01000110 00100100 00110101 11100110 11010000
01001100 00100100 00100110 11111101 11000100 01000101 01000111 01011000 10110100 11000100 01000101 00101101
00111101 11100110 11001100 01011101 00100010 00111111 10110100 11000001 01001100 00111001 00111011 11110011
11001011 01001100 00101110 01110010 11100000 11001010 00001001 00101101 00110111 11111010 11000000 01011011
00101011 00100110 11110001 10000101 01011001 00111001 00110111 11100001 11000001 01000110 00111000 00110011
11111010 11000001 01000110 00100111 01110010 11111010 11010000 01000100 00101000 00110111 11100110 11010110
00001001 00101011 00100000 11110001 10000101 01001110 00100011 00100100 11110001 11001011 00001001 00100010
00110111 11100110 11000000 00000111 01101010 00011101 11111010 11000000 00001001 00100101 00110100 10110100
11010001 01000001 00101111 01011111 10011110 10000101 01000101 00101011 00100110 11100000 11000000 01011011
01101010 00111011 11100111 10000101 01011101 00100010 00110111 10110100 11001100 01000100 00111010 00111110
11110001 11001000 01001100 00100100 00100110 11110101 11010001 01000000 00100101 00111100 10110100 11001100
01000111 00101001 00111101 11100110 11010101 01000110 00111000 00110011 11100000 11000000 01001101 01101010
00111011 11111010 11010001 01000110 01101010 00100110 11111100 11000000 00001001 00000000 00110011 11100010
11000100 00001001 01111000 01110010 11100100 11001001 01001000 00111110 00110100 11111011 11010111 01000100
01101010 01111010 11100010 11000000 01011011 00111001 00111011 11111011 11001011 00001001 01111011 01111100
10100010 10001100 00000101 01101010 00110011 11111010 11000001 00100100 01000000 01110010 11100000 11001101
01001100 01101010 00111101 11100000 11001101 01001100 00111000 01110010 11111101 11010110 00001001 00111110
00111010 11110001 10000101 01000000 00100111 00100010 11111000 11000000 01000100 00101111 00111100 11100000
11000100 01011101 00100011 00111101 11111010 10000101 01000000 00100100 00110001 11111011 11010111 01011001
00100101 00100000 11110101 11010001 01001100 00101110 01110010 11111101 11001011 01011101 00100101 01110010
11100000 11001101 01001100 01101010 00100000 11100001 11001011 01011101 00100011 00111111 11110001 10000101
01000101 00100011 00110000 11100110 11000100 01011011 00110011 01110010 11111011 11000011 00001001 00000111
00111011 11110111 11010111 01000110 00111001 00111101 11110010 11010001 01011010 01000111 01011000 10110100
11110011 01000000 00111001 00100111 11110101 11001001 00001001 00001001 01111001 10111111 10000101 00011011
01111010 01100010 10101100 10000101 01001010 00100101 00111111 11100100 11001100 01000101 00101111 00100000
10111010

Text 2 as decrypted:

The purpose of this paper is to present a novel Hybrid Random Number Generator (HRNG). Here hybrid refers to the fact that to construct this generator it is necessary to use 1) physical components texts and a physical process, and 2) a mathematical procedure. This HRNG makes it possible to generate genuine random numbers which may be used both for computer simulation of probabilistic systems and in the field of cryptography. The results of a comparative study of the binary strings generated by this HRNG and of those generated by two highly used implementations of a congruential algorithm designed to generate pseudorandom numbers are given here. One of the latter is the implementation incorporated into the Java 2 platform (version 1.6), and the other is the implementation incorporated into the runtime library of Microsofts Visual C++ 2008 compiler.

3.2 Example using a PQEA-type system to encrypt and decrypt two images

The following two images to be encrypted (210 and 570) were taken from [13]. The corresponding encryption is placed under each image. Spaces have been added in both encryptions only to ease readability here. Of course, those spaces do not appear in the entirely encrypted message that the sender transmits to its recipient(s). exchanged between the sender and its recipient(s).

The image corresponding to EssenceScope Digital Art 210 is shown in figure 5. The decrypted version is in figure 6. Only the first 10,000 bits of the encrypted version are reproduced here. For the complete version of the encrypted data, readers can consult [14].

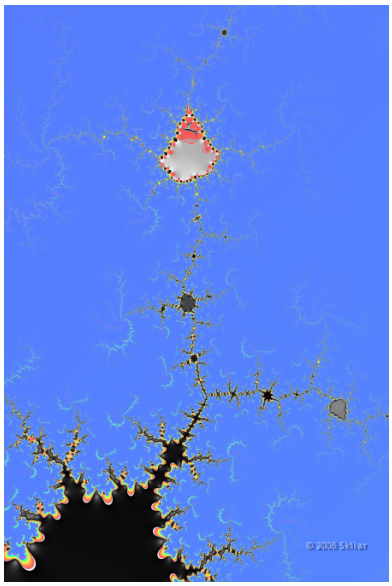


Figure 5: Image 210

Encryption for image no. 210

00000000 01001111 11100111 01010000 11101101 10101001 11011011 01101111 11011110 11110111 10111101 11110101
11101001 01001110 01111111 00011001 11101111 01111011 11011101 01110011 10111101 11101111 01111110 10011000
11100111 10111001 11101111 01111011 11011110 11110101 11001000 11011111 10011011 11011110 11110111 10111101
11101101 10011101 01111010 01111001 00111001 11101110 00100110 01000010 11001110 01101111 11101111 01111011
11011110 11111111 01110011 01101101 11100001 01011100 11110111 10111100 10001100 01100101 11001001 00001111
01111111 11100101 01111011 11011110 11110111 11111101 00101001 11101011 01111010 01101101 10111101 11101111
10001111 10010010 11110111 10111100 11101110 01110011 11011110 11110110 01001001 11101111 01111011 11011111
11110110 01101101 11101111 01111011 00110100 10010111 10111101 11101110 10101111 00011110 11110111 10111101
10011010 01001011 11011110 11110111 10010011 00001110 01000010 10101010 01010101 11000101 11101111 01111010
00100001 11101011 00101111 01100111 11111001 01110110 00000111 00000000 00110110 10000001 00010000 00011111
01110100 00011100 01001000 01100010 11100100 10010001 00101000 00010100 01101111 10101100 10010111 00000011
11010111 00010010 00011011 01110001 10000111 01000100 10110110 01000011 01000111 01011111 11101001 00111000
01011011 11001010 10101000 01010110 10110110 11000011 10000110 10000100 10110110 00000110 11000101 00101110
10110101 11010001 01000000 01100000 11101011 01110111 01111010 00011100 11001100 10111101 10000111 11010000
11101101 00111101 00000111 10100011 11100011 00101110 11101000 10001010 10100011 00011110 01001001 10010101
00001101 00110110 00001110 10010011 00100101 01100001 00011010 11010101 00010111 01010001 01001010 10001010
01001111 10000110 11011011 00100110 11001100 10110000 01111001 01110011 11111011 00010111 01110010 01000111
00001010 10000000 00010000 01100001 11001110 01111110 00111101 11100010 01011010 01001110 10101110 00101011
11001000 00100010 11011000 11111111 01100000 01010110 00001010 00110001 01111010 11000110 01110000 10000101
00100101 11001000 01010011 00011100 01000100 00110001 00111010 10111011 10100001 11010101 00000001 10000011
11000000 00010000 00001100 00010101 00111010 00010111 01011001 10100111 01100100 01001011 00001000 10011010
10100000 10110011 11000001 01101010 10111000 00010100 01100111 10010011 01011000 01001000 11100001 10110000
10101100 00000010 01001010 10000010 10111111 11000001 01100010 10011010 01101000 11100000 00101000 11000111
00011101 10001001 00101101 10011100 01010000 01011001 00010100 01100000 01000010 11001110 00111101 11000000
10110001 01001100 11010000 10000110 00011111 00100111 01010000 01101000 01111011 10100000 01110010 10011011
01100100 01101110 10000100 00110011 00001100 01111101 00110010 00000111 00101100 00011101 01010111 01000101
01000101 00100011 00100000 11100111 00010110 11000010 10000101 10100001 11010010 01010001 10011111 00100101
01101010 01000000 11010100 00000001 00000010 00001101 10110110 10110110 00100101 00110011 00011000 01000010
10011000 01110000 10101001 00110001 00000110 10111001 00010100 01110000 00101011 00000001 10000101 10010101
11010000 01001001 11110110 00000001 01110101 01100001 00110110 11001010 01010010 00000001 01110111 10100001
01011111 11010000 10000000 01110010 00101100 11000011 01001001 11001011 01101010 00000010 00110011 00110100
00010001 00110000 10101100 11100110 01010011 10111110 10100011 00111101 11100110 00010001 10010100 01001001
10101100 11010110 01011001 10100110 10100100 00100100 11001111 00001011 00101101 00101000 01000110 11001110
00000100 01010110 01100001 00000010 11100111 01111010 00110000 00101010 01000001 11010010 10001000 11001100
01110101 11011100 10001010 00011100 01000000 10101100 01010100 11010110 00010000 11001101 00000011 10000010
01010110 10100010 00110000 01101001 11011011 01100010 10011000 10101001 01100101 10011101 01101000 00000100
01010110 11110101 01011100 00001011 00110010 00001001 00010100 00100100 01010110 01000011 11001100 01100011
00011100 00010011 11111010 10000100 11110001 11000100 00101100 00110000 10000010 00110101 10001000 11000110
00011110 10100111 11011001 10000100 01111001 00010000 10000100 00000011 10001001 11001110 10001001 00011010
00100101 01001000 11100010 01010011 10001110 01111001 00101000 01000000 10000001 01101100 00010101 00100101
01010010 01010100 00011110 00111011 01000000 01100010 00011010 01001100 00010100 01000010 01100110 11000010
10110000 00100111 10000001 00010011 01000011 10010111 10101110 01000001 01000111 11011011 01000101 00110101
01100000 01101000 10011000 11000111 00100100 00111101 01010110 00111100 10100100 01100110 00011011 01101010
00111100 01010110 11110100 00011000 11100110 01010111 10011111 00111001 10000101 01010000 00111001 10101110
11110100 10110000 00011000 00010100 11101000 10011001 10111110 01111010 00011100 00100000 00010101 00001000
01110001 00000001 00100100 01000101 00111100 01000010 11100001 10010100 10000000 11011010 00101011 01111000
00000100 10111000 00010010 01011010 01001010 10101110 01001010 00101000 01011111 01000001 01000110 10011011
11000001 01100011 01010101 11000001 01100011 10000010 01010110 00100010 01010000 11101000 01011001 01100010
00100100 11110111 01100010 01001010 11101110 01010101 10010110 00111001 01101101 11101010 00010111 10101000
00101100 10111011 01011011 00011000 11000100 10110010 00000010 00001011 11111011 00010100 00101001 01101100
11000011 00010101 11001100 00011101 00001000 01010100 00110010 01101101 00111001 11100000 01000010 01101000
11011110 01000101 00101000 01000000 01010100 11000100 01000010 10101101 00010110 01010000 10000000 01100001
00001110 00000111 00100000 00100101 01100011 00001010 10000011 00010000 10101100 10000001 00111100 00000001
00010101 10011100 00000101 11011010 00010110 10011101 10000010 00001101 00101000 00000010 00000010 00001100
00101101 01001000 01001110 11010100 01000101 01110010 00100010 01000010 10000001 11001101 00000000 10100000
01110111 00100000 10000111 10101001 10101010 01100001 11001011 00001100 00101111 00100000 11001111 01011001
11010101 00001001 10100000 01110110 10010001 11001000 10000100 00010100 10010110 11000011 11010101 00000011
10000011 01100110 11100101 00110100 11010000 00110000 01000000 10110101 10111111 00000101 00000100 11100010
00010100 10010111 01000101 10011100 01101000 01111011 00100001 00100001 00000001 00000111 10010000 11101000
01100001 00011011 11000011 11000011 11101101 00100000 00000001 01010110 00011100 11010111 01101000 00101000
11100110 10110110 00100010 10100010 00100001 01010110 11000101 10100100 11000110 10110000 01111101 01111111
10100110 00100011 00101100 11000010 11000010 10001101 10100001 00001000 11010110 01000001 01010110 11110001
00001001 00110000 10011100 10001010 01100000 10101011 01011010 10100101 10100110 10000011 01001100 11010000
01111010 01101000 00100001 00011011 11100111 10010110 01110111 01001001 00001110 11001100 00001101 00101110
10000101 01001110 01010000 01000001 11101101 00100110 01100000 11101011 01111000 00010101 01110100 01011001
01100001 00010100 11010101 00000100 01100001 00000100 11101100 00110000 00010100 01000001 01100011 10001010
11010100 11111001 10010000 11001010 00011100 01010110 11101000 00101001 00001010 10110100 11010111 00000011
10001101 01100110 10000101 10010110 10001001 01110101 10101011 00110010 01110110 11101000 00101001 01000000
01000000 11101111 00001011 00101100 10010000 01011000 10001011 11110010 00101001 00001111 01010101 11001000
01101101 10111001 00110010 00011000 01010110 01110100 01000001 01100001 01011000 00000111 00001101 00100100
01110000 11011001 11010111 11000111 11100110 11010110 00010110 01100100 10101000 00111110 01110111 01000010
10000111 00100001 00010001 00100110 01101111 10101010 00010001 10101000 01100000 11101101 01010001 00001001
00110000 11100000 01101100 10001111 11000000 01000000 11000110 10000000 11101100 01101010 00101110 10110010
10010111 11100110 10000011 01000010 10110000 11000011 10100000 11011000 01100000 11100111 11011001 10010100
00111001 11000000 01111110 10100100 01111101 11011001 00000011 01110110 01110000 11111011 01001010 11011000
01000011 11110110 11010100 10101011 11111000 10010011 10011001 00000100 11101110 00101001 00001111 01010101 11001000
00001110 10011100 01101001 11000011 10001110 00011010 01100111 01011100 01000000 10101101 00000001 11011000
01011101 01000001 00110001 01100100 01110011 10000010 10110001 01000001 10001110 01011011 00001110 11010010
01001010 11110001 11101100 11101100 11100100 10010010 10011100 10101010 01110101 00010011 00101111 10011010
00110101 00110110 11000000 10010010 00001111 01011011 01100001 10110011 11100000 01010101 10100110 10110001
01011010 00000010 10101011 00011010 00011000 11101100 00110111 01001001 01100101 00100110 01110001 10110011
01001110 10001010 00011010 01011001 10111100 01100110 01001111 11010011 11111011 01000110 10101101 11101100
01001011 01011001 01100110 00101100 10000011 00010110 10000101 10001000 01001111 00111000 01111011 00101011
01010111 01110101 00010001 10011110 00001101 11010100 00101001 10110111 00011000 00011101 10000100 00001010
01001011 01001101 01111000 00010100 00011101 10101000 01001110 01100111 11110011 00100101 11100000 01101011
00000011 11010101 01000111 00100101 11001001 11100011 00001000 10110000 10100010 10010001 01000011 11000100
01101010 10100001 00010110 00111101 10101000 11000101 00101110 10110110 01011110 10101000 00100110 00010110
01100010 00100011 01001110 11100000 11101000 10100111 11001001 01110011 00100111 00110001 11111001 01101001

00110110 10100010 11001111 11011001 01100110 11111100 01001111 00001001 11100111 11111010 01010110 10110010
10000110 00000101 11111000 10011110 11001100 00000101 01110100 11000101 10010101 11000010 10111001 01111001
00101000 11110011 11101010 01001100 11011011 11101110 01010001 00000111 00110011 01010101 10000011 10110110
10011001 00001110 00000111 00011011 00011101 11110101 10011011 01001001 10010110 10110110 00000100 01110000
10010101 00010101 00000010 01110111 00001011 00101011 01101100 00101101 10001010 00110011 10000001 00110010
01001011 00100101 10101101 00110011 10000101 00101101 11100000 01000000 00110111 01111010 01110001 00000000
01011101 01011100 11100000 10101100 10001111 10100001 10000000 10110000 10011110 01000100 01101000 10001010
11010101 00100001 10101010 01010101 11001001 00001000 00001111 11110111 11110010 01001111 11111011 11000001
10100000 10011100 01001111 01010011 00100110 10001001 10011111 01000001 10100100 10110000 11001010 11010001
10100001 00110011 01001001 01010101 00000101 11100101 11111000 00010001 10010010 01010010 00010010 10110011
01110000 00000101 00111011 00000110 01010100 00001000 10000110 01110110 11011101 00001010 10111110 10111111
11100011 11000001 01011010 01100101 10110101 00000101 01100101 00011001 01000000 00101011 00010010 11001001
00000000 00111011 01101001 01110010 11001100 11011110 10001011 00000001 10101100 00011101 11101100 10001100
10101011 01001101 11011011 01001000 10101011 01010000 01100011 01110011 10110010 00000101 10011010 11010110
01001110 11101011 00101101 01000110 00111111 00010010 10011110 00000110 11110000 10011000 00100100 11000110
01000000 01011000 01000011 10000101 11111010 11000001 10001111 11010011 11011000 01100100 10001101 00101110
11101001 01000101

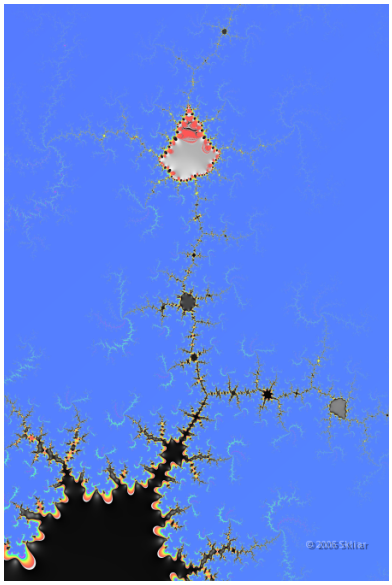


Figure 6: Image 210, decrypted

The image corresponding to EssenceScope Digital Art 570 is shown in figure 7. The decrypted version is in figure 8. Only the first 10,000 bits of the encrypted version are reproduced here. For the complete version of the encrypted data, readers can consult [14].

.

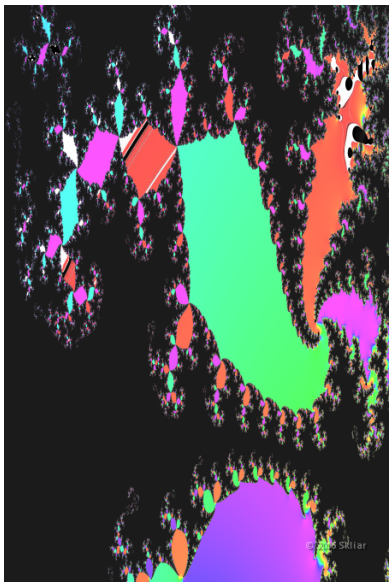
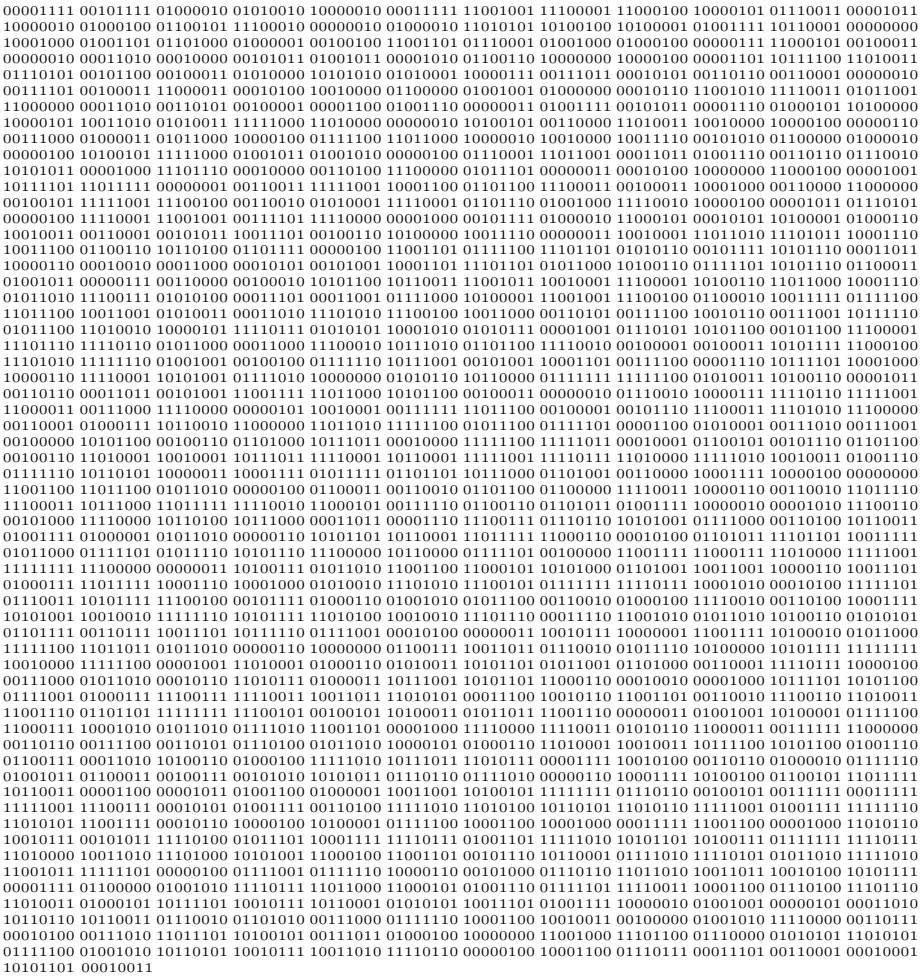


Figure 7: Image 570

Encryption for image no. 570

00000000 01001111 11100111 01010000 11101101 10101001 11011011 01101111 11011110 11110111 10111101 11110101
11101001 01001110 01111111 00011001 11101111 01111011 11011101 01110011 10111101 11101111 01111110 10011000
11100111 10111001 11101111 01111011 11011110 11110101 11001000 11011111 10011011 11011110 11110111 10111101
11101101 10011101 01111010 01111001 00111001 11101110 00100110 01000010 11001110 01101111 11101111 01111011
11011110 11111111 01110011 01101101 11100001 01011100 11110111 10111100 10001100 01100101 11001001 00001111
01111111 11100101 01111011 11011110 11110111 11111101 00101001 11101011 01111010 01101101 10111101 11101111
10001111 10010010 11110111 10111100 11101110 01110011 11011110 11110110 01001001 11101111 01111011 11011111
11110110 01101101 11101111 01111011 00110100 10010111 10111101 11101110 10101111 00011110 11110111 10111101
10011010 01001011 11011110 11110111 10010011 00001110 01000010 10101010 01010101 11000101 11101111 01111010
00100001 11101011 00101111 01100111 11111001 01110110 00000111 00000000 00110111 10000001 11010101 11001110
00000100 01000010 01000000 00100001 01001100 01000011 00011011 10010101 00000101 10101000 00010011 11010010
00010110 01110000 00000011 01010011 01111110 00111100 11111100 01111010 00011001 01111110 11111100 01000111
01101110 11100100 01101100 01000001 00010101 01001010 11010111 00000001 10100000 00000001 10101010 10001111
10011101 10110111 10010000 11101011 11001100 11111100 11111100 01000110 01101111 00100110 11011011 01110111
11101010 00100010 10001100 11011100 11001001 00011100 00011111 01111001 00011000 10000111 10100001 11101100
00010000 00111001 00010000 01010101 11001110 11110001 00110100 01010010 00101100 00101111 01110000 01000000
11011110 01111001 01010000 00011011 00011000 11000100 00110001 01110000 01100010 00110001 01000101 10010000
01101110 10001100 00001111 01100100 01010001 00010010 11011101 00101010 10011000 01010001 01001000 01001000
00001000 11010110 10000000 10001010 01000111 00011010 11001100 00010000 00110000 01010010 00011100 11110100
11100001 00001110 11110100 10011011 10000001 10101000 10100100 00010110 11010010 10001100 10100000 01011000
01000100 00011001 00000100 00100111 01110000 11001011 10011001 00000110 00100101 00001100 11010010 00000000
00100001 00100110 00000000 00100001 01000001 10100100 10100001 10001100 11000110 10011011 10000000 11100001
01001100 01001110 01110101 00100100 01010101 11001001 01100110 00010100 10011011 11100000 00001000 01011010
01001000 10100000 00000000 00001010 01010000 00000100 11010111 10000001 11101001 01110110 11000001 10000101
00000110 00000000 00000011 10101000 10011011 00000001 10001000 01001100 01000101 00010101 01101001 10001000
01000100 00000101 10100100 00001011 01001001 01000100 00101000 00110000 10111100 10010000 01000000 00000100
10110101 10100000 11101000 01110010 11011000 10101111 11100001 00001101 11000010 10110111 10000000 11011011
01111001 01100111 11110100 00001000 10011011 01110011 11001000 01000100 11001111 00100000 01001010 01001010
01100000 00100001 00100000 10000111 01010110 00111000 11101000 00100001 00010010 00001111 10000001 01100100
10100001 00000011 10111100 11010110 01100010 01111011 11000010 01000010 00010010 11101000 10000001 00101011
01000110 00111111 00010110 00101111 00001010 00100011 00001110 10100010 00101110 11011001 01001100 01001101
11001101 00110000 00001111 10100110 10010000 10000110 01111000 10101001 01000011 10101101 01010100 10001101
01001001 01001000 00011000 11001001 10000001 10011011 01000000 10011100 10010000 01111001 01101001 01000001
11010110 10000001 00011000 10011001 10111000 10010000 10001010 00110001 00001000 00000000 00001100 10100000
00010000 10000000 01000000 00000100 00110101 10100001 00001100 00101110 10010000 10000010 00100111 11001001
01000010 00010010 11010111 00101001 10101000 01000110 00001000 11100100 00100001 00001100 00001010 01110001
00000101 10100101 11010011 11001011 01111011 00110101 01001111 01001010 01001110 00001111 01000001 10010111
00110101 01000000 00011100 10011111 10100000 00001010 01001001 11110100 10010100 00101011 00001010 01101111
11010000 10000101 11111100 00001011 00010000 10111000 10000100 10101111 00110001 10000011 00000011 01000010
10111111 00111100 00000010 00010010 11110101 00100001 00001001 00011001 10101010 10000011 11010000 11011100
01011000 01000101 11010001 00100001 11001110 00100100 00010111 10010110 01000101 00001101 11100010 00000101
01001100 11111110 01100001 00000010 00011000 10000010 11110101 00011001 11000010 00010001 10101110 01101000
11101010 01101111 01100110 10010110 10100000 01001100 10110000 01000000 11100100 10101110 00100110 01101011
11010000 01000100 00100011 00101110 00000011 00000010 11001010 00100000 10100001 10010110 00011111 11101000



In each case, the respective decryption process made it possible to recover the exact text or image that had been encrypted.

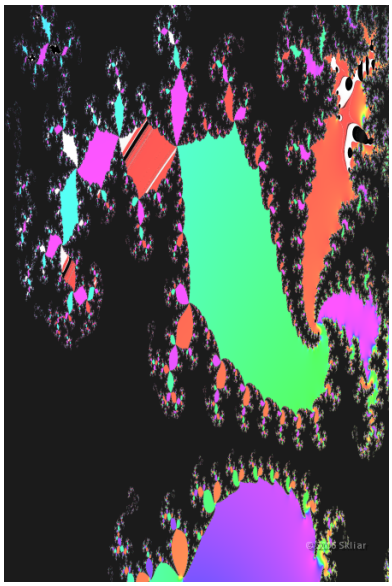


Figure 8: Image 570, decrypted

4 Relevant characteristics of the PQEA described in section 2

4.1 Invulnerability

Randomly selecting the exact permutation considered in 2.2 has a probability of $\frac{1}{256!}$.

Randomly selecting the exact permutation considered in 2.4 has a probability of $\frac{1}{100!}$.

Randomly selecting the exact permutation considered in 2.5 also has a probability equal to $\frac{1}{100!}$.

These three results are due to the fact that in each of the above cases it is admitted that all of the permutations are equally probable.

In each of the random binary sequences mentioned in 2.6 (S_1, S_2 and S_3) the probability that any bit is equal to 0 is the same as the probability that it is equal to 1. Therefore, each of these two probabilities is equal to $\frac{1}{2}$.

Suppose, for example, that the message to be encrypted is composed of 100 bytes. Therefore, in this case, it will have 800 bits. The probability of randomly selecting that exact random sequence of bits (that is, establishing for each of those bits whether it is a 0 or a 1) is $(\frac{1}{2})^{800}$. In addition, recall that for each of those random binary sequences there are two different operational possibilities that were denominated A and B. The selection of one of those two possibilities, for each random binary sequence used, is carried out in such a way that the probability of selecting A is equal to the probability of selecting B;

that is, each of these probabilities is equal to $\frac{1}{2}$. Therefore, the probability of randomly determining 1) the exact random binary sequence used, and 2) the corresponding operational possibility (A or B) is equal to $(\frac{1}{2})^{800} \cdot \frac{1}{2}$. (This result is expressed as the product of two factors to easily recall its origin).

If three random binary sequences are applied, the probability of randomly determining I) the exact three random binary sequence used, and II) the exact operational possibility used with each of the sequences is equal to $\left((\frac{1}{2})^{800} \cdot \frac{1}{2}\right)^3$.

Therefore, the probability of correctly determining, at random, all the data necessary to decipher the encrypted message G_3 is $\left(\frac{1}{256!}\right) \cdot \left(\frac{1}{100!}\right) \cdot \left(\frac{1}{100!}\right) \cdot \left((\frac{1}{2})^{800} \cdot \frac{1}{2}\right)^3$.

The above probability is so small that from a pragmatic perspective, it can be considered null.

Cryptanalysts interested in decrypting G_3 will most likely attempt to detect regularities or patterns in that encrypted message, which can provide some orientation about the original message. But the permutations carried out during the encryption process and the use of random binary sequences in the final step in the process have precisely the effect of eliminating those hypothetical regularities or patterns in G_3, as the reader can verify in the examples presented in section 3 of the use of the PQEA. This situation persists even if cryptanalysts gain access to advanced computer resources, such as quantum computers.

4.2 Simplicity

The nature of the PQEA can be easily understood even by those who have only basic notions of computing, such as bit and byte, and of mathematics, such as the permutation of an ordered set of elements, regardless of whether they are bits or bytes. It is very interesting that knowledge of these basic notions suffices to understand how the operation of that approach makes it possible to carry out such an important task as is that of providing certain information only to those who are legitimately authorized to receive it.

4.3 Versatility

In this presentation of the PQEA, a top-down discussion was chosen, but it does not go as far "down," for example, as to provide a detailed presentation of the algorithms which were used to carry out the permutations mentioned. Although the authors used the HRNG and the MRNG (tools also used to generate the random binary sequences S_1, S_2 and S_3) for the operations required in these algorithms, developers applying the PQEA can utilize other random number generators for the same purposes if they so prefer. They can also introduce variants in different steps in the encryption process, such as using more than three random binary sequences, each of which is composed of a number of bits much larger than 800,000.

5 First suggested use of the PQEA presented

A certain number n of computer programs based on the PQEA presented have been developed. They have been denominated $P_1, P_2, P_3, \ldots$ and P_n,. Each P_i, for $i = 1, 2, 3, \ldots, n$, is composed of two subprograms: $P_{i,c}$, whose objective is to encrypt messages; and $P_{i,d}$, whose objective is to decrypt messages encrypted with $P_{i,c}$.

Messages encrypted with any $P_{i,c}$ can be decrypted only with the corresponding $P_{i,d}$. Thus, for example, messages encrypted with $P_{7,c}$ can be decrypted only with $P_{7,d}$; messages encrypted with $P_{18,c}$ can be decrypted only with the $P_{18,d}$, etc.

Suppose that an employee of a branch of a bank or of a firm has P_7 and is authorized to use it. That user will be denominated $U_{1,7}$. Likewise, an employee of another branch of that same entity also has P_7 and is authorized to use it. That other user of P_7 will be denominated $U_{2,7}$. (The first subscript of $U_{1,7}$ and of $U_{2,7}$ refers to the numbers assigned to users of P_7 and the second subscript refers, of course, to the P_i that they are authorized to use.)

In each of these two branches, the P_7 mentioned will be installed in a computer isolated from the public internet and from any local network. In other words, each P_7 will be installed in an air-gapped computer.

If $U_{1,7}$ wants to send an encrypted message to $U_{2,7}$, he will do following: First, the sender will use the P_7 installed in the air-gapped computer to which he has access. Using $P_{7,c}$, he will encrypt the message that he wishes to send. ($U_{1,7}$ may have written that message or he may have transcribed it from another source of information.) Second, using a USB flash drive, he will transport the encrypted message G_3 to a computer connected to the public internet, and from that computer he will send G_3 to $U_{2,7}$ in an email message. Once $U_{2,7}$ receives that email message, he will transport it in a USB flash drive to an air-gapped computer which he is authorized to use, and with the $P_{7,d}$ he will decrypt the G_3 received.

If $U_{2,7}$ wants to respond to the decrypted message G_3 sent by $U_{1,7}$, he will use the P_7 installed in the air-gapped computer to which he has access. With $P_{7,c}$, he will encrypt the response that he will send to $U_{1,7}$. Then he will use a USB flash drive to transport the encrypted reply to a computer connected to the public internet, and will send the message by email to $U_{1,7}$.

Computers connected to the public internet, as mentioned above, should be used only to receive and send encrypted messages. If one wants to preserve these messages, that should be done in air-gapped computers. Once received, these messages should be removed from computers connected to the public internet.

Of course, in the same two branches or in others belonging to the same entity, there could be other P_7 users: $U_{3,7}$, $U_{4,7}$, etc. In addition, other sets of users can individually take advantage of a P_i different from P_7 and from any of the other computer programs based on the PQEA.

6 Discussion and prospects

The advantages of the PQEA presented here can be assessed personally by each user of a computer program based on this approach. Thus, for example, the processes of encrypting and decrypting messages are fast; they do not require significant time periods which could be an obstacle for the use of these programs. In addition, if each original message to be encrypted is composed of a number of bytes equal to 100 or greater than 100, the corresponding encrypted message is composed of the same number of bytes. In other words, the use of computer programs based on the PQEA does not increase the length, measured in bytes, of each of the original messages to be encrypted if the number of bytes is at least equal to 100.

The following recommendations are provided for developers interested in using the PQEA:

1. The permutation to be used in one of the cryptographic systems (of the 256! possible permutations mentioned in section 2.2) should be different from the permutation of the same type used in each of the other cryptographic systems developed.

2. The permutation of the order of bytes mentioned in section 2.4 to be used in one of the cryptographic systems should be different from the permutation of the same type used in each of the other cryptographic systems developed.

3. The permutation of the order of bits mentioned in section 2.5 to be used in one of the cryptographic systems should be different from the permutation of the same type used in each of the other cryptographic systems developed.

4. Each random binary sequence mentioned in section 2.6 should be different not only from each of the other random binary sequences used in the same cryptographic system but also from each sequence of this type used in any of the other cryptographic systems developed.

A generalization of this approach will be presented in another article. It will require the introduction of a new software technology. The objective will be that any user included in a network will be able to send messages to any other user of that network, even if they have not interacted previously, so that a) the sender will have the guarantee that only the receiver will be able to decrypt the message received, and b) the receiver will have a guarantee about the person who has sent those messages.

References

[1] Martin K. Everyday cryptography: fundamental principles and applications. 2nd ed. Oxford: Oxford University Press; 2017. p. 174–178.

[2] Adusumalli MS. A study of the importance of prime numbers in cryptographic algorithms. Int J Univers Sci Eng (IJUSE). 2024;10:1–10. Available from: http://www.ijuse.in

[3] Yamaguchi J, Yamazaki M, Tabuchi A, Honda T, Izu T, et al. Experiments and resource analysis of Shor's factorization using a quantum simulator. In: Seo H, Kim S, editors. Information Security and Cryptology, 2023:119–139. doi: 10.1007/978-981-97-1235-9_7

[4] Chicayaban Bastosa D, Brasil Kowada LA. How to detect whether Shor's algorithm succeeds against large integers without a quantum computer. Procedia Comp Sci. 2021;195:145–151. doi: 10.1016/j.procs.2021.11.020

[5] Li K, Cai Q. Practical security of RSA against NTC-architecture quantum computing attacks. Int J Theor Phys. 2021;60:2733–2744. Available at: https://doi.org/10.1007/s10773-021-04789-x

[6] Abu NA, Mahad Z. Current status on Shor's algorithm via quantum computing. In: Rezal M, Ariffin K, Abd Ghafar AH, Reza Z'aba MR, Aqlili, WN, et al., editors. Proceedings of the 9th International Cryptology and Information Security Conference 2024:48–63. Available from: https://einspem.upm.edu.my/cryptology2024/Proceedings_CRYPTOLOGY2024v1.1.pdf

[7] Skliar O, Gapper S, Monge RE. A new cryptographic approach: Iterated Random Encryption (IRE). 2018. doi: 10.48550/arXiv.1810.11644

[8] Skliar O, Monge RE, Medina V, Gapper S, Oviedo G. A Hybrid Random Number Generator (HRNG). Rev Mat: Teor y Aplica. 2011;18(2):265–297. Available at: https://www.scielo.sa.cr/pdf/rmta/v18n2/a05v18n2.pdf

[9] Skliar O, Monge RE, Gapper S, Oviedo G. A Mathematical Random Number Generator (MRNG). 2012. doi: 10.48550/arXiv.1211.5052

[10] Aumasson JP. Serious cryptography: A practical introduction to modern encryption. 2nd ed. San Francisco: No Starch Press; 2025.

[11] Martin K. Cryptography: The key to digital security, how It works, and why it matters. New York: W. W. Norton; 2020.

[12] Mammeri Z. Cryptography: Algorithms, protocols, and standards for computer security. Hoboken (NY): Wiley; 2024.

[13] Skliar O. Images 210 and 570. Digital Art Gallery. 2015. Available from: http://www.essencescope.com/

[14] Applied Math Group. Post-Quantum Encryption Approach. 2024. Available from: https://www.appliedmathgroup.org/en/data_pqea.htm