electronics

Article

A Study on the Simple Encryption of QR Codes Using Random Numbers

Iori Okubo [1], Seiya Ono [1], Hyun-Woo Kim [1], Myungjin Cho [2,*,†] and Min-Chul Lee [1,*,†]

[1] Graduate School of Computer Science and Systems Engineering, Kyushu Institute of Technology, 680-4 Kawazu, Iizuka-shi 820-8502, Fukuoka, Japan; okubo.iori855@mail.kyutech.jp (I.O.); ono.seiya343@mail.kyutech.jp (S.O.); kim.hyunwoo547@mail.kyutech.jp (H.-W.K.)
[2] School of ICT, Robotics, and Mechanical Engineering, Hankyong National University, IITC, 327 Chungang-ro, Anseong 17579, Kyonggi-do, Republic of Korea
* Correspondence: mjcho@hknu.ac.kr (M.C.); lee@csn.kyutech.ac.jp (M.-C.L.)
† These authors contributed equally to this work.

Abstract: Recently, with the widespread adoption of quick response (QR) code payments, there have been incidents of unauthorized use of QR codes presented at the time of payment, due to theft or duplication. As a countermeasure, conventional QR code payment systems are designed to update the QR code periodically. However, there is a problem: it is possible to make a payment using an illegally obtained QR code until the update. Therefore, it is necessary to encrypt the QR code itself to prevent its duplication. The objective of this research is to prevent fraudulent use of QR payments by combining image encryption using random numbers and Rivest Cipher 4 (RC4). In this paper, we perform image encryption using random numbers generated from a uniform distribution for QR codes presented at the time of payment and encrypt the seed value, which is the decryption key, using RC4. As a result, the proposed encryption method prevents unauthorized use of the QR code used for payment by stealing the image, and the processing speed and encryption strength are sufficient. Histogram analysis, key sensitivity analysis, and correlation coefficients were used to measure encryption strength. Finally, the proposed method is expected to enable more secure use of QR payments compared to conventional systems.

Keywords: QR code encryption; QR payment security; random number

Citation: Okubo, I.; Ono, S.; Kim, H.-W.; Cho, M.; Lee, M.-C. A Study on the Simple Encryption of QR Codes Using Random Numbers. *Electronics* **2024**, *13*, 3003. https://doi.org/10.3390/electronics13153003

Academic Editor: Srinivas Sampalli

Received: 7 June 2024
Revised: 27 June 2024
Accepted: 29 July 2024
Published: 30 July 2024

1. Introduction

Quick response (QR) code payments [1–3] have become popular due to their ease of use. Along with this trend, there are incidents of unauthorized use of QR codes [4–8] at the time of payment. There are two main types of QR code payment methods: the user-scan method, where the store displays a QR code for the user to scan, and the store-scan method, where the store scans a QR code presented by the user. In the user-scan method, there are confirmed cases where QR codes posted by stores are switched and money is transferred to the third party. In the store scan method, there are confirmed cases where QR codes are stolen or duplicated while payment is prepared, and payment is made using the duplicated code. In this study, we consider the case of the store scan method only. In the current QR code payment system, the QR code is updated after a certain period of time as a countermeasure against unauthorized use by theft or duplication. However, it is necessary to encrypt the QR code itself because payment may be made with an illegally obtained QR code until the QR code is updated. Various studies have been conducted on image encryption, such as double random phase encryption (DRPE) [9–12]. DRPE is a secure and fast encryption method that uses two random phase masks, and the complex conjugate of the second random phase mask can be used as the key information for decryption. However, since DRPE encryption uses the Fourier transform, the encrypted image is a complex number. Since the complex number consists of real and imaginary parts, it is

difficult to display the encrypted image as a single image. Thus, it makes DRPE unsuitable for QR code payment systems. To enhance the security without making unnecessary changes in the QR code payment system, it requires that the encrypted image be displayed as a single image. In addition, the QR code recognition rate after decryption must be 100% to avoid problems that would make QR code payments impossible. Moreover, the processing speed of encryption and decryption must be fast so that the advantage of quick payment is not lost. Based on these requirements, it is necessary to construct an image encryption system for the QR code payment that can display the encrypted image as a single image, has a 100% recognition rate of QR codes after decryption, and has a high processing speed. In this paper, we propose an encryption system that achieves these requirements by combining random numbers generated from a uniform distribution [13,14] based image encryption and Rivest Cipher 4 (RC4) [15–18]. Image encryption using these random numbers poses a risk that the decryption key can be illegally obtained during sharing. As a countermeasure, RC4, a type of symmetric key cryptography, is used to encrypt the decryption key to increase security.

2. Principle

2.1. QR Code

A QR code is a type of two-dimensional bar code developed to record more information than a one-dimensional bar code and is a technology that can encode a variety of data, including URLs and text. QR is an abbreviation for "Quick Response" indicating that the contents of the code can be read quickly. While a one-dimensional bar code can only record the information horizontally, a QR code can record the information both vertically and horizontally in two dimensions, allowing more information to be recorded in a smaller space. Therefore, numbers as well as alphabetical information can be recorded. A QR code also has Reed–Solomon code error correction, which enables it to be read even if the code is dirty or damaged. There are four levels of error correction in a QR code—the higher the level, the higher the error correction capability. Table 1 shows the error correction levels and error correction capability [19].

Table 1. Level of error correction in QR code.

Error Correction Level	Approximate Correction Capacity
L	7%
M	15%
Q	25%
H	30%

Levels L and M in QR codes are commonly used in general environments, while levels Q and H are recommended for environments where QR codes are likely to be contaminated, such as in factories. The structure of the QR code is illustrated in Figure 1.

Finder patterns shown in Figure 1 are the symbols used to detect the positions of QR codes, which are located in the upper left, upper right, and lower left of the QR codes. Timing patterns are the symbols needed to correct the overall distortions when reading the QR codes and consist of alternating white and black cells in fixed positions. The alignment pattern is the symbol used to compensate for the distortion of each cell that occurs when the QR code is read from the angle and is located in the lower-left corner of the QR code. The format information contains details needed to determine the level of the error correction function described earlier. In this paper, encryption using random numbers generated from a uniform distribution will be performed for each pixel of the QR code.

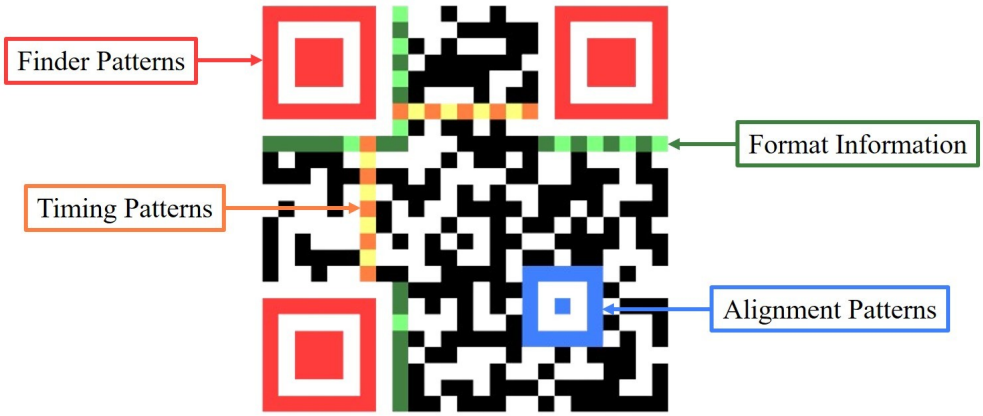


Figure 1. Structure of the QR code.

2.2. Uniform Distribution

Uniform distribution is a probability distribution in which all events occur with equal probability. It can be classified into discrete-type uniform distribution and continuous-type uniform distribution. Discrete-type uniform distribution is used when the random variable X is discrete, while continuous-type uniform distribution is used when the random variable X is continuous. If the random variable X follows a discrete uniform distribution, the probability $X = k$ is expressed as follows [20]:

$$P(X = k) = \frac{1}{N}(k = 1, \ldots, N). \tag{1}$$

In this case, N is a natural number, and the expected value $E(X)$ and the variance $V(X)$ are expressed as follows:

$$E(X) = \frac{N-1}{2}, \tag{2}$$

$$V(X) = \frac{N^2-1}{12}. \tag{3}$$

On the other hand, when the random variable X follows a continuous-type uniform distribution for $a \leq X \leq b$, the probability density function is expressed as follows [20]:

$$f(x) = \frac{1}{b-a}\ (a \leq X \leq b), who \tag{4}$$

$$f(x) = 0\ (X < a, X > b). \tag{5}$$

In this case, the expected value $E(X)$ and variance $V(X)$ are expressed as follows [20]:

$$E(X) = \frac{a+b}{2}, \tag{6}$$

$$V(X) = \frac{(b-a)^2}{12}. \tag{7}$$

The uniform distribution is one of the most fundamental and widely used probability distributions. Therefore, in this paper, encryption is performed using random numbers generated from the continuous-type uniform distribution.

2.3. Rivest Cipher 4 (RC4)

RC4 is a type of symmetric key cipher. Common-key cryptography can be divided into two types: stream ciphers and block ciphers. A stream cipher encrypts the data sequentially in bits or bytes while a block cipher encrypts data in batches of a specific size. This feature enables the stream cipher to encrypt and decrypt the data faster than the block cipher. RC4 is classified as the stream cipher. It encrypts the data by generating a keystream and obtaining an exclusive OR of the keystream and the plaintext. Figure 2 illustrates the procedure of the RC4 process.

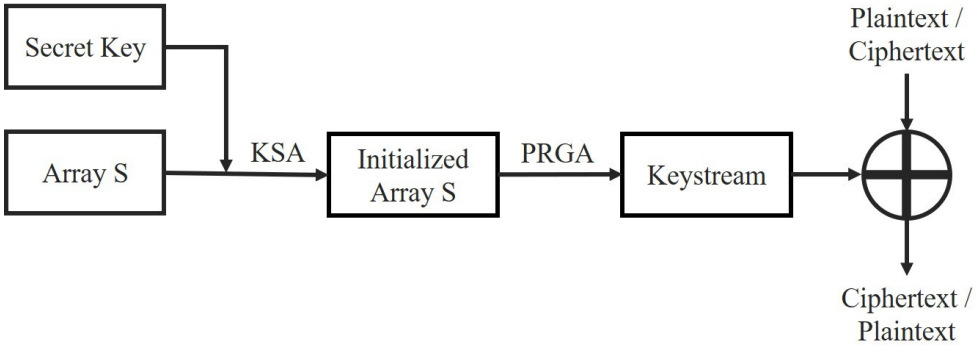


Figure 2. Procedure of the RC4 process.

RC4 process first defines an array S, where S[0], S[1], . . ., and S[255] are filled with values between 0 and 255 in that order. S is then initialized with the secret key using the key-scheduling algorithm (KSA). The pseudo-code for the KSA process is shown in Algorithm 1 [21].

Algorithm 1 KSA

$j = 0$
for i = 0 to 255 **do**
  $S[i] = i$
end for
for i = 0 to 255 **do**
  $j = (j + S[i] + K[i])\ mod\ 256$
  swap S[i] and S[j]
end for

Next, the keystream is generated from the initialized array S using the pseudo-random generation algorithm (PRGA). The pseudo-code for the PRGA process is shown in Algorithm 2 [21].

Algorithm 2 PRGA

$i = 0$
$j = 0$
for K = 0 to N − 1 **do**
  $i = (i + 1)\ mod\ 256$
  $j = (j + S[i])\ mod\ 256$
  swap S[i] and S[j]
  $K = S[(S[i] + S[j])\ mod\ 256]$
end for

Finally, the ciphertext is generated by obtaining the exclusive OR of the generated keystream and the plaintext. During decryption, the keystream is generated in the same manner, and the ciphertext is decrypted back to plaintext by applying the exclusive OR of the ciphertext and the keystream. In this paper, RC4 is used as the decryption key for image encryption using uniform random numbers. Since QR codes are updated at a certain time

in QR payment services, RC4 is chosen for its processing speed, enabling quick transactions at the time of payment rather than for its cryptographic strength.

3. Simple Encryption of QR Codes Using Random Numbers

3.1. System Overview

In this system, encryption is performed on the QR code using random numbers generated from the uniform distribution, and the seed value, which is the decryption key, is encrypted using RC4. The overview of this system is illustrated in Figure 3.

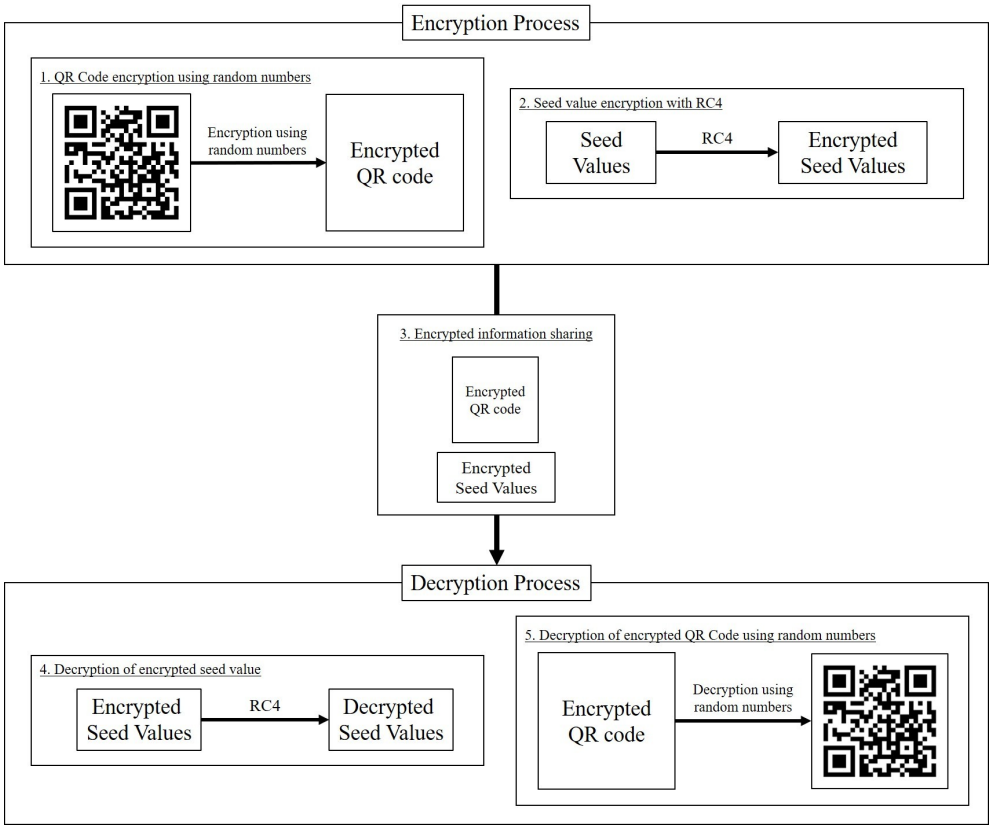


Figure 3. Overview of the proposed system.

QR code encryption using random numbers.

In this paper, the QR code encryption scheme using random numbers generated from a uniform distribution is proposed. As the encryption procedure, five mask images are generated using these random numbers and added to the QR code. By making the pixel values of the mask image integers in the range of 0 to 254, the QR code pixel values after encryption take the values of 0 to 255. In this encryption, the seed value required to generate the mask image is used as the decryption key. The details are described in Section 3.2.

Seed value encryption using RC4

As described in Section 2.3, RC4 generates the keystream using secret key processes, such as KSA and PRGA, and encrypts by obtaining an exclusive OR with the plaintext. It is applied to encrypt the seed value, which is the decryption key in QR code encryption using random numbers generated from a uniform distribution.

Sharing of encrypted information.

QR code encrypted with random numbers is read with the camera and the decryption key encrypted using RC4 is shared.

Decryption of encrypted seed values.

Encrypted seed values are decrypted using RC4. As with encryption, the keystream

is generated using the secret key, and the seed values can be decrypted by obtaining an exclusive OR with the encrypted seed value.

Decryption of encrypted QR codes using random numbers.

The mask image, which is the homogeneous mask image used for encryption, is generated by specifying the decrypted seed value in the random number generated from the uniform distribution. Decryption is performed by subtracting this mask image from the encrypted QR code.

3.2. Image Encryption Using Random Numbers Generated from a Uniform Distribution

In image encryption using random numbers, encryption is performed by adding five mask images with a QR code. In the initial process, five images of the same size as the QR code to be encrypted are generated using a random number with a specified seed value. If the seed values used to generate five mask images are the same, five mask images will be equal. This means that adding five mask images together will not increase security. Therefore, the fixed value is added to the seed value specified when generating each mask image, and five different mask images can be generated by shifting the seed value. In this paper, the first mask image is encrypted using the specified seed value. When generating the other four mask images, fixed increments—95, 154, 483, and 753—are added to the respective seed values for encryption. Five mask images generated in this way are added together to form a single mask image. The largest pixel value in the mask image is then set to 254, ensuring that the pixel value of the encrypted image, after adding the mask image to the QR code, covers 256 shades. Figure 4 presents how mask images are generated.

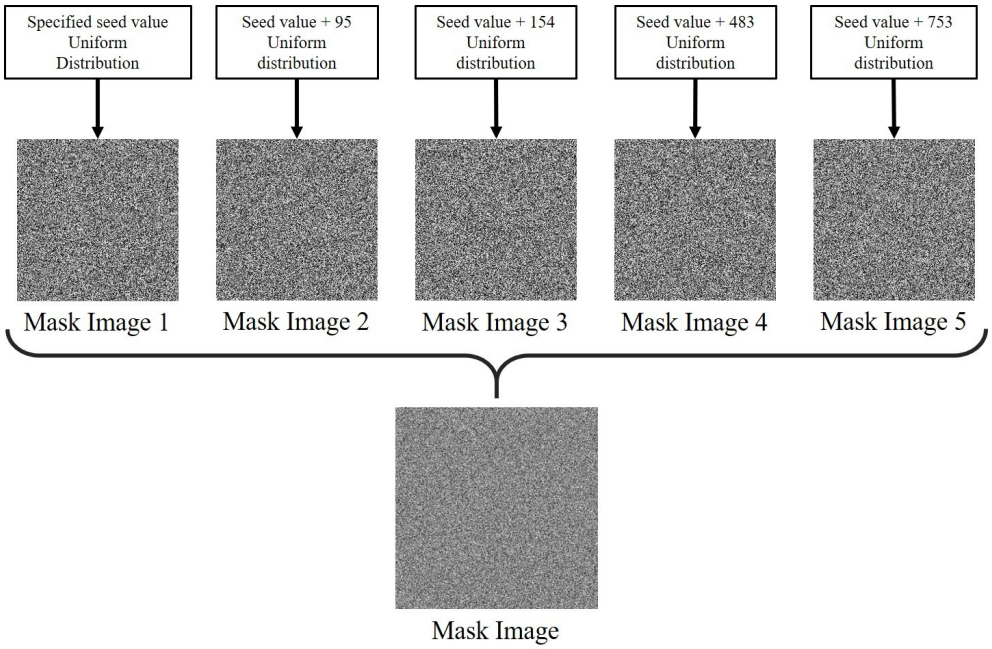


Figure 4. Generation of mask images using random numbers from the uniform distribution.

The pseudo-code for the process of generating a mask image is shown in Algorithm 3.

The generated mask image is added to the image for encryption. Figure 5 shows the encryption result using the proposed method.

In the decryption process, the mask image, which is the homogeneous mask image used for encryption, can be generated by sharing the seed value. Decryption can be performed by subtracting the generated mask image from the encrypted image. Figure 6 shows the decryption result by the proposed method.

Algorithm 3 Generation of mask images.

1: $[Ny, Nx] = size(OriginalQRcode)$
2: $key = EncryptionKey$
3: $rng(key)$
4: $MaskImage1 = rand([Ny, Nx])$
5: $rng(key + 95)$
6: $MaskImage2 = rand([Ny, Nx])$
7: $rng(key + 154)$
8: $MaskImage3 = rand([Ny, Nx])$
9: $rng(key + 483)$
10: $MaskImage4 = rand([Ny, Nx])$
11: $rng(key + 753)$
12: $MaskImage5 = rand([Ny, Nx])$
13: $MaskImage = MaskImage1 + MaskImage2 + MaskImage3 + MaskImage4 + MaskImage5$
14: $MaskImage = MaskImage./max(max(MaskImage)).*254$

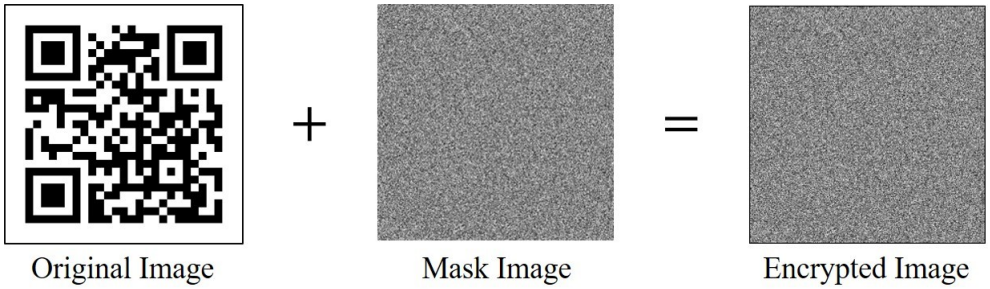


Figure 5. QR code encryption using random numbers from a uniform distribution.

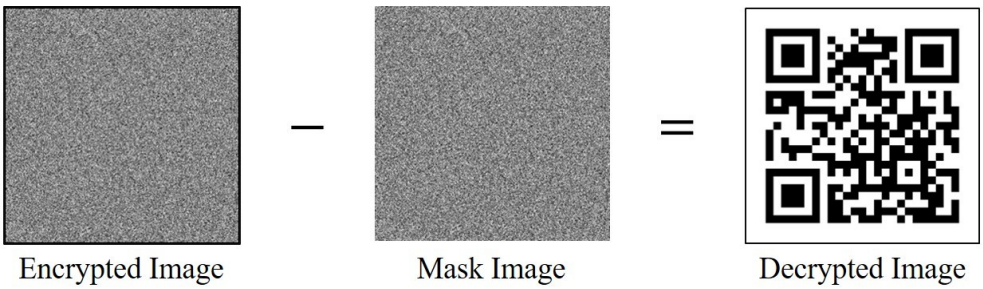


Figure 6. QR code decryption using random numbers from a uniform distribution.

3.3. Encryption of Decryption Key Using RC4

In image encryption using random numbers from a uniform distribution, to generate the same mask image on both the encryption and decryption sides, the seed value of the Poisson random number used to generate the mask image is shared as the decryption key. However, if the seed value is shared as is and both the encrypted QR code and the seed value are obtained by a malicious third party, there is a risk of easy decryption. Therefore, it is required that the seed value is encrypted using RC4. However, when RC4 is used for encryption, the encrypted value may be converted into complex symbols or Greek letters. In such a case, the rapid sharing of decryption keys necessary for QR payments will be affected. Therefore, the proposed method restricts the numerical values for quick QR code payments.

4. Experimental Procedure and Results

In this section, we describe the method and results of the verification experiment of the proposed system. In this research, 100 types of QR codes are prepared for experiments, and encryption and decryption using the proposed method are performed for each of them.

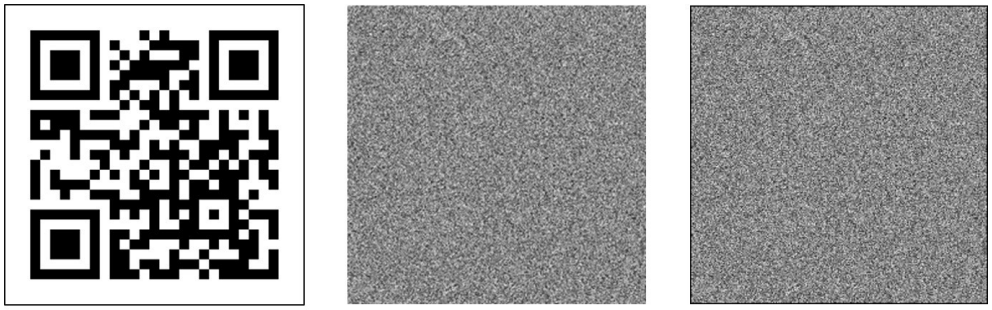
4.1. Experimental Procedure

In this experiment, we fixed the size of the QR code to be encrypted as 240 px × 240 px, which is the recommended size for printed materials. First, the QR code was encrypted using the proposed method and displayed on a computer screen. Then, a screenshot of the displayed QR code after encryption was taken. This allowed the experiment to be conducted in an environment that was not affected by reflections on the screen or noise from surrounding light sources, which would occur when taking photographs. Next, to verify the decryption process, we performed decryption using the proposed method on a screenshot to check the recognition rate. Histogram analysis, key sensitivity analysis, and correlation coefficients were used to assess the encryption strength. In addition, we measured the processing time required for encryption and decryption.

4.2. Results

4.2.1. Encryption of QR Codes Using Random Numbers Generated from a Uniform Distribution

Figure 7 shows the QR codes to be encrypted, the mask image generated by setting the seed value to 42, and the encryption result by adding the mask image and QR code.



(a) Original Image (b) Generated Mask Image (c) Encrypted Image

Figure 7. QR code encryption procedure using random numbers generated from a uniform distribution. (**a**) Original image, (**b**) generated mask image, and (**c**) encrypted image.

The encrypted QR code shown in Figure 7c may not be recognized as an actual QR code.

Experiments were conducted on the encryption of decryption keys using RC4. Table 2 shows the results of encrypting with RC4 for five different seed values.

Table 2. Encryption of the seed value using RC4.

Seed Value	Encrypted Seed Value
42	111
2	71
318	379
8375	8434
23,537	23,476

From Table 2, all the seed values were changed to different values using RC4.

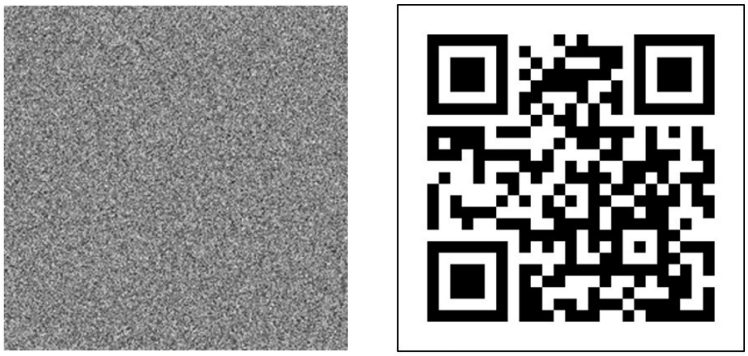
4.2.2. Decryption of QR Codes Using Random Numbers Generated from a Uniform Distribution

Table 3 summarizes the results of decrypting the encrypted seed values as shown in Table 2 using RC4.

Table 3. Decryption of the seed value using RC4.

Encrypted Seed Value	Decrypted Seed Value
111	42
71	2
379	318
8434	8375
23,476	23,537

From Table 3, it is confirmed that the original seed value was decrypted. Figure 8a shows the mask image generated using the decrypted seed value. Figure 8b shows the result of decryption by subtracting the mask image in Figure 8a from the screenshot of the encrypted image, as shown in Figure 7c.



(a) Generated Mask Image (b) Decrypted Image

Figure 8. QR code decryption procedure using random numbers generated via uniform distribution. (**a**) Generated mask image and (**b**) decrypted image.

Figure 8 shows that encryption using a uniform distribution is capable of complete decryption. The recognition rate was 100% for 100 different QR codes used in the experiment.

Table 4 shows the measured processing time required to encrypt an image using random numbers generated by uniform distribution.

Table 4. Time required for encryption by uniform distribution.

Seed Value	QR1	QR2	QR3
42	0.3701 [s]	0.3325 [s]	0.3346 [s]
318	0.3607 [s]	0.3394 [s]	0.3397 [s]
23,537	0.3781 [s]	0.3302 [s]	0.3643 [s]

Table 4 shows that the processing time required for encryption is not affected by the seed value or QR code to be encrypted and that the processing speed is always sufficient for QR payments.

Table 5 shows the results from measuring the processing time required for decryption.

Table 5. Time required for decryption via uniform distribution.

Seed Value	QR1	QR2	QR3
42	0.3381 [s]	0.3558 [s]	0.3452 [s]
318	0.3256 [s]	0.3454 [s]	0.3365 [s]
23,537	0.3448 [s]	0.3480 [s]	0.3463 [s]

Table 5 shows that the processing time required for decryption is not affected by the seed value or QR code to be decrypted and that the processing speed is always sufficient for QR settlements.

In the currently proposed method, the fixed values—95, 154, 483, and 75—are added to avoid the same seed value specified when generating the mask image in the encryption process. Randomly determining this fixed value is expected to improve the security strength. However, when introducing this function, the fixed values must be equal, as adding different fixed values on the encryption and decryption sides will make decryption impossible.

4.2.3. Security Analysis for Encryption Using Random Numbers Generated from Uniform Distribution

To measure the encryption strength, a histogram analysis is performed. The histogram of the encrypted image shown in Figure 7c is shown in Figure 9.

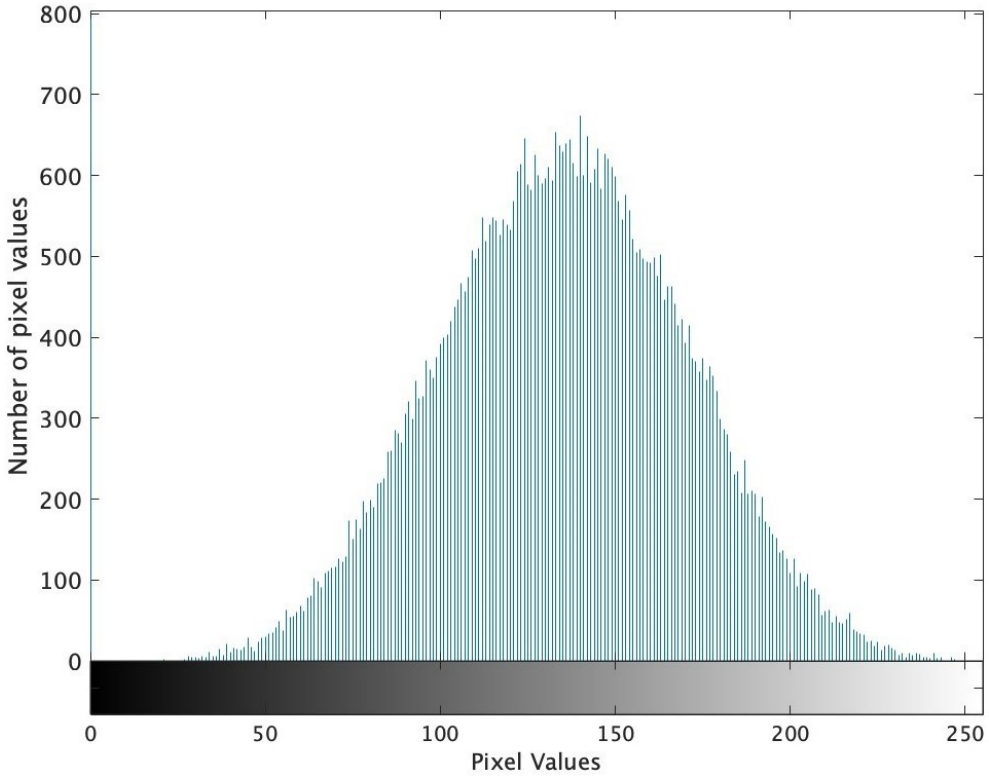


Figure 9. Histogram of the encrypted image.

The closer the histogram of the encrypted image is to uniform, the better it is as an encryption because it indicates that the pixel values are randomly distributed. Figure 9 shows that the histogram of the encrypted image by the proposed method follows a normal distribution. This is because normalization is performed during the encryption process by the proposed method. If the histogram is not uniform, it is known to be vulnerable to known-plaintext and chosen-plaintext attacks.

Next, a key sensitivity analysis was conducted. A key sensitivity analysis is a method of analyzing the impact of a small change in a cryptographic key on the encrypted image. The images used for the key sensitivity analysis are shown in Figure 10. Figure 10a shows the original image, Figure 10b shows the encrypted image with the seed value set to 41, Figure 10c shows the encrypted image with the seed value set to 42, and Figure 10d shows the encrypted image with the seed value set to 43.

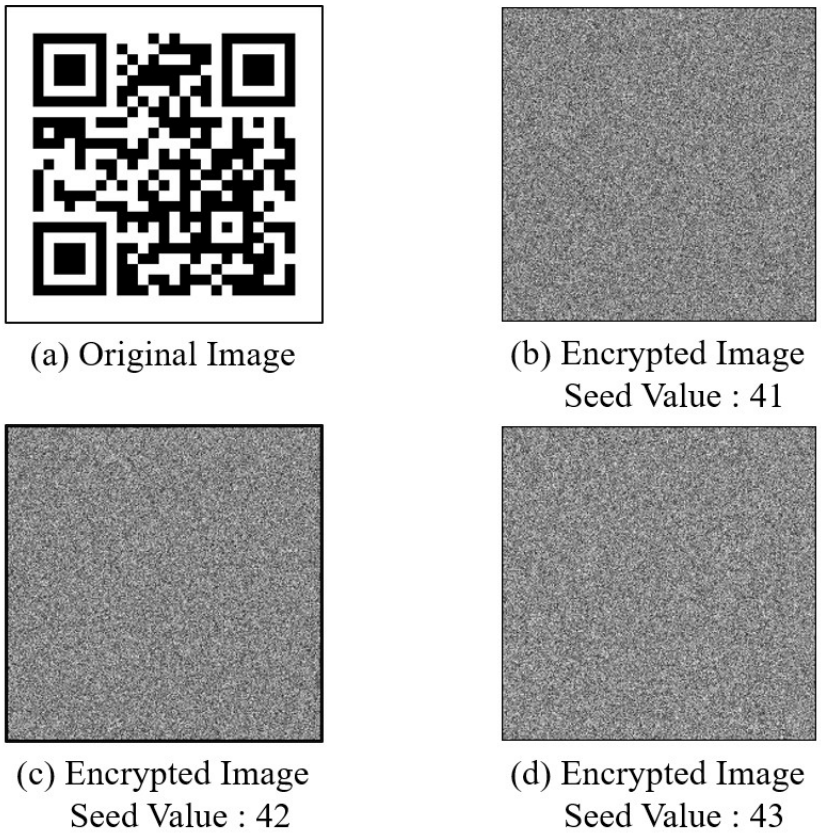


Figure 10. Images used for key sensitivity analysis. (**a**) Original image, (**b**) image encrypted with a seed value of 41, (**c**) image encrypted with a seed value of 42, and (**d**) image encrypted with a seed value of 43.

Key sensitivity is analyzed by comparing the encrypted images shown in Figure 10b–d. The correlation coefficient of the corresponding pixel values between each image was used as a method of comparison. The results are summarized in Table 6.

Table 6. Time required for decryption by uniform distribution.

Encrypted Image 1	Encrypted Image 2	Correlation Coefficient
Figure 10b	Figure 10c	0.19881
Figure 10b	Figure 10d	0.20005
Figure 10c	Figure 10d	0.20054

Correlation coefficients below 0.2 indicate that there is little correlation. The results in Table 6 show that the proposed method is capable of completely different encryption when the encryption key differs even slightly. The results of decrypting an image encrypted with a seed value of 42 using a different seed value are shown in Figure 11. Figure 11a shows the image decrypted with the seed value set to 42. Figure 11b shows the image decrypted with the seed value set to 41. Figure 11c shows the image decrypted with the seed value set to 43. Figure 11d shows the image decrypted with the seed value set to 90.

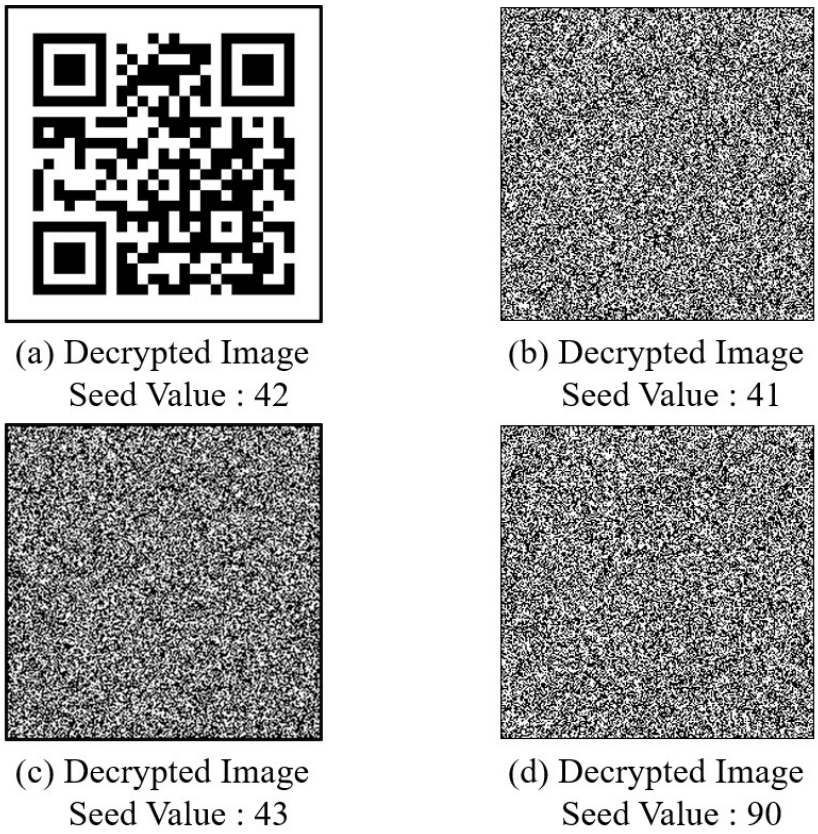


Figure 11. Decryption with a seed value different from the seed value used for encryption. (**a**) Original image, (**b**) image decrypted with a seed value of 41, (**c**) image decrypted with a seed value of 42, and (**d**) image decrypted with a seed value of 43.

From Figure 11, decryption with a value that differs even slightly from the seed value used during encryption is impossible. From the results in Figures 10 and 11, it can be said that the proposed method is key-sensitive and an excellent encryption method.

The correlation between two adjacent pixel values in the encrypted image was determined to analyze the resistance to a statistical attack [22,23], which is a type of attack method on encrypted images. A statistical attack is an attack method that analyzes the statistical characteristics of an encrypted image to deduce information about the encryption algorithm and key. As part of the analysis procedure, first, 1000 pairs of horizontally adjacent pixels are extracted from the encrypted image. For each pair, one pixel value is denoted as x and the other as y, and we use Equation (8) to obtain the correlation coefficient r [24–26].

$$r = \frac{\frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})^2}\sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \bar{y})^2}} \tag{8}$$

In this case, n is the data number, which in this case is 1000. In addition, x_i and y_i are the individual values of x and y, respectively, $\bar{x}$ is the average of x, and $\bar{y}$ is the average of y. The correlation coefficient between two horizontally adjacent pixel values in the encrypted image obtained using Equation (8) is 0.05367. As an interpretation of correlation coefficients, it is known that the closer the absolute value of the correlation coefficient is to 1, the stronger the correlation and that there is almost no correlation when the value is between 0.0 and 0.2. This indicates that there is almost no correlation between two horizontally adjacent pixels in the encrypted image by the proposed method. From the encryption point of view, the lower the correlation coefficients, the more resistant to statistical attacks. Therefore, in addition to the horizontal direction, the correlation coefficients for two adjacent pixels in

the vertical and diagonal directions were determined, respectively. The results are shown in Table 7, where the correlation coefficients of two adjacent pixel values in the horizontal, vertical, and diagonal directions are shown for comparison.

Table 7. Correlation coefficient of two adjacent pixel values.

	Horizontal	Vertical	Diagonal
Encrypted Image	0.05367	0.08145	0.01468
Original Image	0.88693	0.90088	0.82142

From Table 7, the correlation coefficient is reduced by encryption using the proposed method. This shows that the encrypted image has no correlation between each pixel value and is resistant to statistical attacks.

A comparison is made between encryption using the proposed method and existing encrypted QR codes. Existing encrypted QR codes do not encrypt the QR codes themselves, but rather the data before they are converted into QR codes. The process of generating an encrypted QR code first involves encrypting the data using a symmetric key cipher such as AES. Next, the encrypted data are converted into a QR code. When reading the QR code, the encrypted data are obtained by reading the QR code. The data are then decrypted using the previously shared key. When comparing the proposed method with existing encrypted QR codes, the proposed method has the advantage of visual encryption, which makes it resistant to forgery and tampering of the encrypted image. However, as existing encrypted QR codes use AES, the encryption strength of existing encrypted QR codes is superior. However, as QR codes are updated at certain times in QR payments, the encryption strength of the encryption by the proposed method is considered sufficient.

It has been confirmed that the QR code sizes used for QR payments are in the range of 200 px × 200 px~400 px × 400 px. Therefore, experiments were also conducted for QR code sizes of 200 px × 200 px, 300 px × 300 px, and 400 px × 400 px. The processing times required to encrypt QR codes of 200 px × 200 px, 300 px × 300 px, and 400 px × 400 px are shown in Tables 8–10, respectively.

Table 8. The processing time required to encrypt a 200 px × 200 px QR code.

Seed Value	QR1	QR2	QR3
42	0.3663 [s]	0.3552 [s]	0.3639 [s]
318	0.3608 [s]	0.3735 [s]	0.3372 [s]
23,537	0.3413 [s]	0.3619 [s]	0.3540 [s]

Table 9. The processing time required to encrypt a 300 px × 300 px QR code.

Seed Value	QR1	QR2	QR3
42	0.3667 [s]	0.3788 [s]	0.3773 [s]
318	0.3754 [s]	0.4090 [s]	0.3828 [s]
23,537	0.3793 [s]	0.3773 [s]	0.3901 [s]

Table 10. The processing time required to encrypt a 400 px × 400 px QR code.

Seed Value	QR1	QR2	QR3
42	0.3880 [s]	0.3901 [s]	0.3728 [s]
318	0.3947 [s]	0.3949 [s]	0.3949 [s]
23,537	0.3947 [s]	0.4005 [s]	0.4077 [s]

Tables 8–10 show that the processing times required for encryption increase slightly as the image sizes increase. However, for QR codes of the same size, the processing speed

is sufficient for QR payments and is not affected by the seed value used for encryption or the type of QR code. Next, the processing times for decryption for each size are shown in Tables 11–13.

Table 11. The processing time required to decrypt a 200 px × 200 px QR code.

Seed Value	QR1	QR2	QR3
42	0.3428 [s]	0.3489 [s]	0.3492 [s]
318	0.3556 [s]	0.3607 [s]	0.3434 [s]
23,537	0.3646 [s]	0.3496 [s]	0.3549 [s]

Table 12. The processing time required to decrypt a 200 px × 200 px QR code.

Seed Value	QR1	QR2	QR3
42	0.3555 [s]	0.3636 [s]	0.3637 [s]
318	0.3639 [s]	0.3842 [s]	0.3644 [s]
23,537	0.3730 [s]	0.3541 [s]	0.3683 [s]

Table 13. The processing time required to decrypt a 200 px × 200 px QR code.

Seed Value	QR1	QR2	QR3
42	0.3979 [s]	0.3919 [s]	0.3957 [s]
318	0.3824 [s]	0.3955 [s]	0.3826 [s]
23,537	0.3938 [s]	0.3829 [s]	0.3927 [s]

Tables 11–13 show that the processing times required for decryption and encryption increase slightly as the image sizes increase. However, for QR codes of the same size, the processing speed is sufficient for QR payments without being affected by the seed value used for encryption or the type of QR code. The correlation coefficients between two neighboring pixels in the horizontal, vertical, and diagonal directions were determined in the encrypted images of 200 px × 200 px, 300 px × 300 px, and 400 px × 400 px QR codes with the seed value set to 42. The results are summarized in Table 14.

Table 14. Correlation coefficient of two adjacent pixels of each size.

	Horizontal	Vertical	Diagonal
200 px × 200 px	0.09448	0.19338	0.03013
300 px × 300 px	0.08762	0.14641	0.00888
400 px × 400 px	0.06314	0.08098	0.00275

The correlation coefficient is less than 0.2 and hardly correlated, indicating that the proposed method is resistant to statistical attacks regardless of the size of the QR code to be encrypted.

5. Conclusions

5.1. Outcome

In this paper, we developed a system for encrypting QR codes by combining random numbers generated from a uniform distribution or Poisson random numbers and RC4. In the conventional QR payment system, unauthorized use of QR codes for payment has occurred due to the theft of QR codes, but our proposed system is one step closer to preventing unauthorized use of QR codes due to theft. When random numbers generated from a uniform distribution are used in the proposed method, the average time required for encryption is 0.350 s and the average time required for decryption is 0.343 s, while when Poisson random numbers are used, the average time required for encryption is 0.663 s and the average time required for decryption is 0.655 s. These results indicate that QR codes

can be encrypted with sufficient encryption strength without losing the advantage of QR payment, which allows for quick payment.

5.2. Prospects for the Future

On the other hand, since the decryption process is performed on a screenshot of the encrypted image in our experiments, the noise caused by screen reflections and ambient light sources that occur when the QR code is read is not taken into account. To incorporate the proposed system into QR payment, it is essential to design a filter to remove external noise, which will be a future challenge. In addition, the value after encryption using RC4 is only limited to numbers so that the encrypted decryption key can be shared quickly, which makes the system vulnerable to brute-force attacks. Including alphabets and symbols in the encrypted value would increase the encryption strength, but would affect the advantage of QR payments, which is the speed of payment. Therefore, improving the encryption method for the decryption key, while balancing encryption strength and rapid sharing, will be a future challenge. Next, RC4 was used to encrypt the decryption key due to its high processing speed, but various vulnerabilities have been reported in RC4. A future task will be to verify how the vulnerability affects this system, as well as to verify the use of other encryption methods. Finally, in the experimental environment of this work, the average time required for encryption was 0.350 s and the average time required for decryption was 0.343 s. However, when incorporating the system into the QR payment system, it must be considered that the system will be incorporated into a device with lower computing power than a computer, so the possibility of a decrease in processing speed should also be considered.

Author Contributions: Conceptualization, I.O. and S.O.; validation, H.-W.K.; data curation, I.O.; writing—original draft preparation, I.O.; writing—review and editing, M.C.; supervision, M.-C.L. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by JSPS KAKENHI grant number 24K01120.

Data Availability Statement: Data are contained within the article.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. De Luna, I.R.; Liébana-Cabanillas, F.; Sánchez-Fernández, J.; Muñoz-Leiva, F. Mobile payment is not all the same: The adoption of mobile payment systems depending on the technology applied. *Technol. Forecast. Soc. Chang.* **2019**, *146*, 931–944. [CrossRef]
2. Nseir, S.; Hirzallah, N.; Aqel, M. A secure mobile payment system using QR code. In Proceedings of the 2013 5th International Conference on Computer Science and Information Technology, Amman, Jordan, 27–28 March 2013; pp. 111–114.
3. Liébana-Cabanillas, F.; De Luna, I.R.; Muñoz-Leiva, F.J. User behaviour in QR mobile payment system: The QR Payment Acceptance Model. *Technol. Anal. Strateg. Manag.* **2015**, *27*, 1031–1049. [CrossRef]
4. Kieseberg, P.; Leithner, M.; Mulazzani, M.; Munroe, L.; Schrittwieser, S.; Sinha, M.; Weippl, E. QR code security. In Proceedings of the 8th International Conference on Advances in Mobile Computing and Multimedia, Paris, France, 8–10 November 2010; pp. 430–435.
5. Narayanan, A.S. QR codes and security solutions. *Int. J. Comput. Sci. Telecommun.* **2012**, *3*, 69–72.
6. Soon, T.J. QR code. *Synth. J.* **2008**, 2008, 59–78.
7. Law, C.Y.; So,S. QR codes in education. *J. Educ. Technol. Dev. Exch. (JETDE)* **2010**, *3*, 7. [CrossRef]
8. Krombholz, K.; Frühwirt, P.; Kieseberg, P.; Kapsalis, I.; Huber, M.; Weippl, E. QR code security: A survey of attacks and challenges for usable security. In *Human Aspects of Information Security, Privacy, and Trust: Proceedings of the Second International Conference, HAS 2014, Held as Part of HCI International 2014, Heraklion, Crete, Greece, 22–27 June 2014*; Proceedings 2; Springer International Publishing: Berlin/Heidelberg, Germany, 2014; pp. 79–90.
9. Refregier, P.; Javidi, B. Optical image encryption based on input plane and Fourier plane random encoding. *Opt. Lett.* **1995**, *20*, 767–769. [CrossRef] [PubMed]
10. Jang, J.Y.; Inoue, K.; Lee, M.C.; Cho, M. Information authentication of three-dimensional photon counting double random phase encryption using nonlinear maximum average correlation height filter. *J. Opt. Soc. Korea* **2016**, *20*, 228–233. [CrossRef]
11. Cho, K.O.; Lee, M.C.; Cho, M. Three-dimensional photon counting double-random-phase encryption with occlusion layers. *J. Inf. Disp.* **2014**, *15*, 127–133. [CrossRef]

12. Honda, K.; Lee, J.; Kim, H.W.; Cho, M.; Lee, M.C. Enhanced security computational double random phase encryption by using additional random function. In Proceedings of the 2021 International Conference on Information and Communication Technology Convergence (ICTC), Jeju Island, Republic of Korea, 20–22 October 2021; pp. 155–159.
13. Matsumoto, M.; Nishimura, T. Mersenne twister: A 623-dimensionally equidistributed uniform pseudo-random number generator. *Acm Trans. Model. Comput. Simul. (TOMACS)* **1998**, *8*, 3–30. [CrossRef]
14. Kuipers, L.; Niederreiter, H. *Uniform Distribution of Sequences*; Courier Corporation: North Chelmsford, MA, USA, 2012.
15. Mousa, A.; Hamad, A. Evaluation of the RC4 algorithm for data encryption. *Int. J. Comput. Sci. Appl.* **2006**, *3*, 44–56.
16. Fluhrer, S.; Mantin, I.; Shamir, A. Weaknesses in the key scheduling algorithm of RC4. In Proceedings of the 8th Annual International Workshop, SAC 2001, Toronto, ON, Canada, 16–17 August 2001; Revised Papers 8; Springer: Berlin/Heidelberg, Germany, 2001; pp. 1–24.
17. Klein, A. Attacks on the RC4 stream cipher. *Des. Codes Cryptogr.* **2008**, *48*, 269–286. [CrossRef]
18. Paul, G.; Maitra, S. *RC4 Stream Cipher and Its Variants*; CRC Press: Boca Raton, FL, USA, 2011.
19. Tiwari, S. An introduction to QR code technology. In Proceedings of the 2016 International Conference on Information Technology (ICIT), Bhubaneswar, India, 22–24 December 2016; pp. 39–44.
20. Bendat, J.S.; Piersol, A.G. *Random Data: Analysis and Measurement Procedures*; John Wiley & Sons: Hoboken, NJ, USA, 2011.
21. Siahaan, A.P.U. An overview of the RC4 algorithm. *IOSR J. Comput. Eng. (IOSR-JCE)* **2017**, *18*, 67–73.
22. Westfeld, A.; Pfitzmann, A. Attacks on steganographic systems: Breaking the steganographic utilities EzStego, Jsteg, Steganos, and S-Tools-and some lessons learned. In Proceedings of the International Workshop on Information Hiding, Dresden, Germany, 29 September–1 October 1999; Springer: Berlin/Heidelberg, Germany, 1999; pp. 61–76.
23. Chen, G.; Mao, Y.; Chui, C.K. A symmetric image encryption scheme based on 3D chaotic cat maps. *Chaos Solitons Fractals* **2004**, *21*, 749–761. [CrossRef]
24. Taylor, R. Interpretation of the correlation coefficient: A basic review. *J. Diagn. Med. Sonogr.* **1990**, *6*, 35–39. [CrossRef]
25. Schober, P.; Boer, C.; Schwarte, L.A. Correlation coefficients: Appropriate use and interpretation. *Anesth. Analg.* **2018**, *126*, 1763–1768. [CrossRef] [PubMed]
26. Akoglu, H. User's guide to correlation coefficients. *Turk. J. Emerg. Med.* **2018**, *18*, 91–93. [CrossRef] [PubMed]