Computer Science and Engineering 2016, 6(2): 19-24
DOI: 10.5923/j.computer.20160602.01

# A New Method for Ciphering a Message Using QR Code

**Sawsan K. Thamer, Basheer N. Ameen***

Computer Science Department, College of Science, Al-Nahrain University, Iraq

**Abstract** In this paper, we have introduced a new data-hiding algorithm, where message is converted to QR code (Quick Response Code) and generate QR for mask (Key). QR Codes are mainly used to carry or store messages because they have higher or large storage capacity than any other normal conventional ‘barcodes’. In the present work the authors have introduced the encryption technique by XORing part (series of bits) of QR message with the same part of QR mask (key) to encrypt any message and then embedding the key into the resulted QR. The resulted QR code may be sent to destination or may be saved for future use. In this encryption method authors have used bit-manipulation, byte-reshuffling and generalized this method. The ciphering method used here has been tested on different plain texts and it was found that the method is unbreakable using traditional cryptanalysis techniques like frequency analysis, plain-text attack, Differential attack, Brute-force attack, etc. The data is encrypted using a symmetric key method, then inserted in QR code, so that data cannot be easily retrieved without adequate authorization / permission.

**Keywords** QR code, Cryptography, Security

## 1. Introduction

In today’s world, security is a big issue and securing important data is very essential, so that the data cannot be intercepted or misused for any kind of unauthorized use. The hackers and intruders are always ready to get through personal data or important data of a person or an organization, and misuse them in various ways. For this reason, the field of cryptography is very important and the cryptographers are trying to introduce new cryptographic methods to secure the data as much as possible. Keep his valuable data like passport information, bank statements, social security number, etc. with himself/herself all the time, but he/she is always afraid of doing so because this information are threatened and can be easily intercepted by outsiders for misuse. We choose another example; a bank manager wants to instruct his subordinates about the process of a huge transaction. If this data is not encrypted properly, then it can be retrieved by a hacker to reverse the transaction process to credit a different account. For this reason, encryption of data and hiding data from unauthentic usage is very important. This problem can be solved by encrypting the data and hiding it in a QR Code [1-3], which can be kept with the person all the time and the QR Code scanner with a software, using the method in this paper, can be used to decode with the authentic password the information saved in it.

* Corresponding author:
basheer_nahiz@yahoo.com (Basheer N. Ameen)
Published online at http://journal.sapub.org/computer

## 2. QR Code

QR code (abbreviated from Quick Response Code) is the trademark for a type of *matrix barcode* (or two-dimensional *barcode*) first designed for the *automotive industry in Japan*. A barcode is a machine-readable optical label that contains information about the item to which it is attached. Four standardized encoding modes (numeric, alphanumeric and byte/binary) could be stored as QR for efficient data store.

The QR Code system became popular outside the automotive industry due to its fast readability and greater storage capacity compared to standard *UPC barcodes*. Applications include product tracking, item identification, time tracking, document management, and general marketing. [4]

A QR code consists of black modules (square dots) arranged in a square grid on a white background, which can be read by an imaging device (such as a camera, scanner, etc.) and processed using *Reed–Solomon* error correction until the image can be appropriately interpreted. The required data are then extracted from patterns that are present in both horizontal and vertical components of the image. [4]

### 2.1. Design

Unlike the older, one-dimensional barcodes that were designed to be mechanically scanned by a narrow beam of light, a QR code is detected by a 2-dimensional digital *image sensor* and then digitally analyzed by a programmed processor. The processor locates the three distinctive squares at the corners of the QR code image, using a smaller square (or multiple squares) near the fourth corner to normalize the image for size, orientation, and angle of viewing. The small dots throughout the QR code are then converted to binary numbers and validated with an error-correcting algorithm.

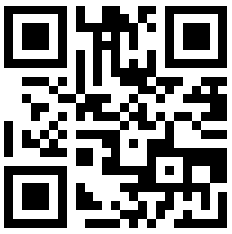
## 2.2. Storage

The amount of data that could be stored in the QR code symbol depends on the data type (mode, or input character set), version (1, …, 40, indicating the overall dimensions of the symbol), and error correction level. The maximum storage capacities occur for 40-L symbols (version 40, error correction level L) as shown in Table (1) [5, 6]:

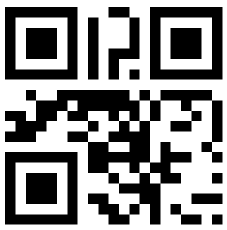
**Table 1.** Maximum character storage capacity (40-L)

| Input mode | max. characters | bits/char | possible characters, default encoding |
|---|---|---|---|
| Numeric only | 7,089 | 3⅓ | 0, 1, 2, 3, 4, 5, 6, 7, 8, 9 |
| Alphanumeric | 4,296 | 5½ | 0–9, A–Z (upper-case only), space, $, %, *, +, -, ., /, : |
| Binary/byte | 2,953 | 8 | ISO 8859-1 |

Here are some samples of QR code symbols:



**Version 1 (21×21). Content:"Ver1"**



**Version 2 (25×25). Content: "Version 2"**

**Version 3 (29×29). Content: "Version 3 QR Code"**

**Version 4 (33×33). Content: "Version 4 QR Code, up to 50 char"**

## 2.3. Error Correction

Codewords are 8 bits long and use the Reed–Solomon error correction algorithm with four error correction levels. The higher the error correction level, the less storage capacity. The following table lists the approximate error correction capability at each of the four levels as declared in table (2):

**Table 2.** Errors correction levels & it's storage capacity

| | |
|---|---|
| Level L (Low) | 7% of codewords can be restored. |
| Level M (Medium) | 15% of codewords can be restored. |
| Level Q (Quartile)[7] | 25% of codewords can be restored. |
| Level H (High) | 30% of codewords can be restored. |

In larger QR symbols, the message is broken up into several Reed–Solomon code blocks. The block size is chosen so that at most 15 errors can be corrected in each block; this limits the complexity of the decoding algorithm. The code blocks are then interleaved together, making it less likely that localized damage to a QR symbol will overwhelm the capacity of any single block.

Due to error correction, it is possible to create artistic QR codes that still scan correctly, but contain intentional errors to make them more readable or attractive to the human eye, as well as to incorporate colors, logos, and other features into the QR code block [8, 9].

It is also possible to design artistic QR codes without reducing the error correction capacity by manipulating the underlying mathematical constructs [10, 11].

## 2.4. Encoding

The format information records two things: the error correction level and the mask pattern used for the symbol. Masking is used to break up patterns in the data area that might confuse a scanner, such as large blank areas or misleading features that look like the locator marks. The mask patterns are defined on a grid that is repeated as necessary to cover the whole symbol. Modules corresponding to the dark areas of the mask are inverted. The format information is protected from errors with a BCH code, and two complete copies are included in each QR symbol that is show in figure (1). [12]

BCH code is the abbreviation for (Bose-Chaudhuri-Hochquenghem) *code*. A multilevel, cyclic, *error*-correcting, variable- length *digital* code used to correct errors up to approximately 25% of the total number of digits. Note: BCH codes are not limited to *binary* codes, but may be used with multilevel *phase-shift keying* whenever the number of levels is a prime number or a power of a prime number, such as 2, 3, 4, 5, 7, 8, 11, and 13. A BCH code in 11 levels has been used to represent the 10 decimal digits plus a sign *digit*.

The message dataset is placed from right to left in a zigzag pattern, as shown in figure (2). In larger symbols, this is complicated by the presence of the alignment patterns and the use of multiple interleaved error-correction blocks, as shown in figure (3).

Four-bit indicators are used to select the encoding mode and convey other information as shown in table (3). Encoding modes can be mixed as needed within a QR symbol.

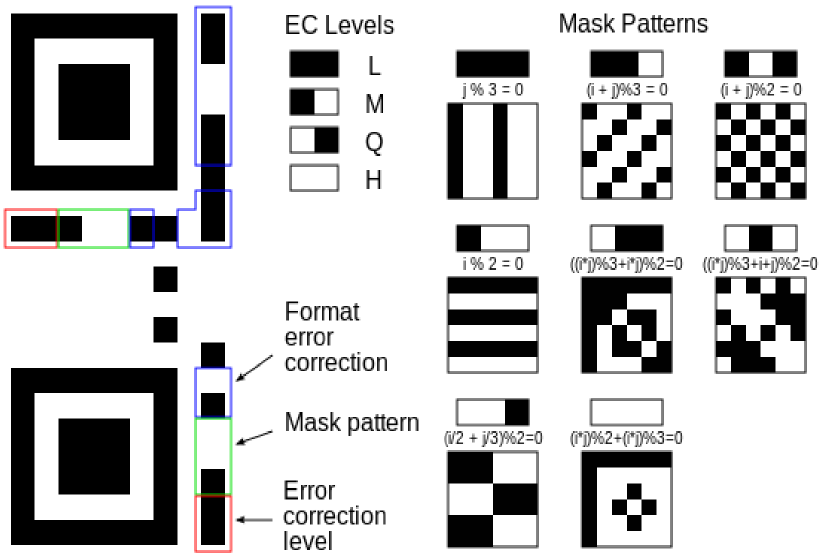


**Figure 1.** Meaning of format information in QR

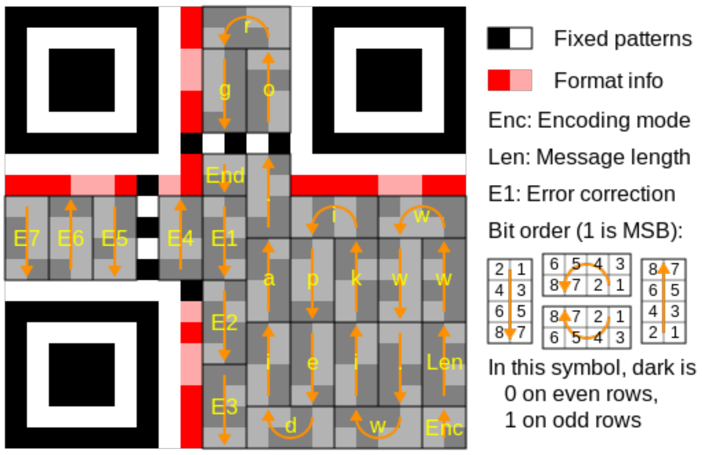


**Figure 2.** Message placement within a QR

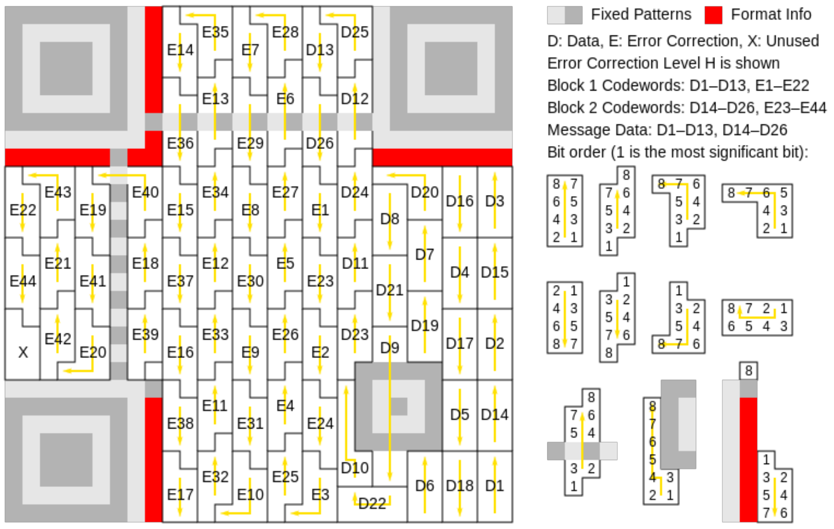

Fixed Patterns
Format Info
D: Data, E: Error Correction, X: Unused
Error Correction Level H is shown
Block 1 Codewords: D1–D13, E1–E22
Block 2 Codewords: D14–D26, E23–E44
Message Data: D1–D13, D14–D26
Bit order (1 is the most significant bit):

**Figure 3.** Larger symbol illustrating interleaved

**Table 3.** Encoding modes

| Indicator | Meaning |
|---|---|
| 0001 | Numeric encoding (10 bits per 3 digits) |
| 0010 | Alphanumeric encoding (11 bits per 2 characters) |
| 0100 | Byte encoding (8 bits per character) |
| 0000 | End of message |

After every indicator that selects an encoding mode is a length field that tells how many characters are encoded in that mode. The number of bits in the length field depends on the encoding and the symbol version as shown in table (4).

**Table 4.** Number of bits per length field

| Encoding | Ver. 1–9 | 10–26 | 27–40 |
|---|---|---|---|
| Numeric | 10 | 12 | 14 |
| Alphanumeric | 9 | 11 | 13 |
| Byte | 8 | 16 | 16 |

Alphanumeric encoding mode stores a message more compactly than the byte mode can, but cannot store lower-case letters and has only a limited selection of punctuation marks, which are sufficient for rudimentary web addresses as shown in table (5). Two characters are coded in an 11-bit value by this formula: $V = 45 \times C1 + C2$

**Table 5.** Alphanumeric character codes

| Code | Character | Code | Character | Code | Character | Code | Character | Code | Character |
|---|---|---|---|---|---|---|---|---|---|
| 00 | 0 | 09 | 9 | 18 | I | 27 | R | 36 | Space |
| 01 | 1 | 10 | A | 19 | J | 28 | S | 37 | $ |
| 02 | 2 | 11 | B | 20 | K | 29 | T | 38 | % |
| 03 | 3 | 12 | C | 21 | L | 30 | U | 39 | * |
| 04 | 4 | 13 | D | 22 | M | 31 | V | 40 | + |
| 05 | 5 | 14 | E | 23 | N | 32 | W | 41 | – |
| 06 | 6 | 15 | F | 24 | O | 33 | X | 42 | . |
| 07 | 7 | 16 | G | 25 | P | 34 | Y | 43 | / |
| 08 | 8 | 17 | H | 26 | Q | 35 | Z | 44 | : |

## 3. Proposed Algorithm

In our approach, new data-hiding algorithm is introduced. Where, two QR codes are generated one for the message and the other for the Key. Then, the two QRs are XORed in a specific part to encrypt the message. Then the key is embedded inside the resulted QR Code.

### 3.1. Algorithm of Convert Plaintext to QR_code

```
Step 1: write message (text).
Step 2: generate QR code for the message.
Step 3: save QR image as P.
```

### 3.2. Algorithm of Convert Key to QR_code

```
Step 1: write key as numbers or text.
Step 2: generate QR code for the key.
Step 3: save QR image as K.
```

### 3.3. Algorithm of get Begin Indies of Data Area in QR

```
Step 1: start.
Step 2: Do loop to get beginning of data area in plain.bmp with width i as height j.
Step 3: end.
```

### 3.4. Algorithm of Encryption

```
Step 1: start.
Step 2: load QR image P.
Step 3: load QR image k.
Step 4: define cipher as bitmap file with dimensions width (wd) & height (hg).
Step 5:Call function to put P(0)(0) to P(i)(j) in cipher(0)(0) to cipher(i)(j).
Step 6: loop statement x=i
loop statement y=j
cipher(x)(y)=P(x)(y) XOR k(x)(y)
next y,x.
Step7: end.
```

### 3.5. Algorithm of Putting Key in Cipher Bitmap File

```
Step 1: start.
Step 2: binarization each character or number in key as 8bit.
Step 3: loop statement l<key (length)
    If (key(l)=255)
        Key(l)=254;
    Else
        Key(l)=1;
   End if.
           Next l
Step 4: if statement
    If (key(l)=255)
        Key(l)=253;
   Else
        Key(l)=2;
   End if.
Step 5: end.
```

### 3.6. Algorithm of Getting Key from Cipher Bitmap File

```
Step 1: start.
Step 2: loop statement until key(l)=253 or 2
     If (key(l)=254 or 1)
        Str=concat(str,'1');
     Else
        Str=concat(str,'0');
   End if
```

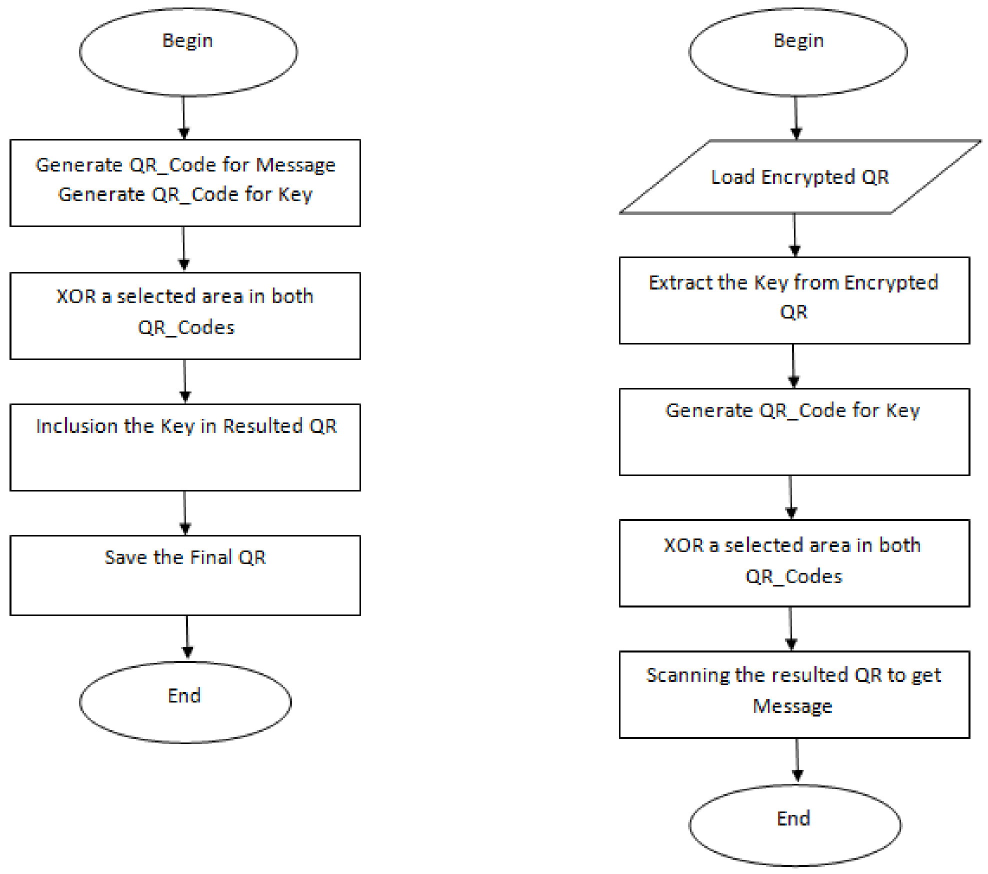

Begin
Generate QR_Code for Message
Generate QR_Code for Key
XOR a selected area in both QR_Codes
Inclusion the Key in Resulted QR
Save the Final QR
End
Begin
Load Encrypted QR
Extract the Key from Encrypted QR
Generate QR_Code for Key
XOR a selected area in both QR_Codes
Scanning the resulted QR to get Message
End

**Figure 4.** Encryption and decryption processes

```
    Next l
Step 4: collect each 8bit in str and get character of this collection.
Step 5: end.
```

### 3.7. Algorithm of Decryption

```
Step 1: start.
Step 2: load QR image c.
Step 3: get width (wd) & height (hg) of c.
Step 4: define plain as bitmap file with dimensions width (wd) & height (hg).
Step 5:Call function to put cipher(0)(0) to cipher(i)(j) in plain(0)(0) to plain(i)(j).
Step 6: loop statement x=i
            loop statement y=j
            plain(x)(y)=cipher(x)(y) XOR key(x)(y)
   next y,x.
Step 7: end.
```

Figure (4) summarizes the encryption and decryption processes.

## 4. Testing and Result

The test includes two processes encryption process and decryption process.

### 4.1. Encryption Process

For example, choose a message “book” and a key “1”. Figure (5) illustrates the QRs for message, key & the resulted QR or final QR image. The final QR is unreadable.

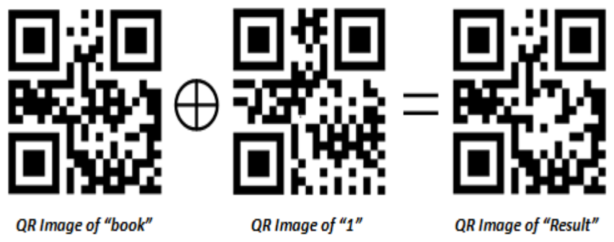


**Figure 5.** Encryption process

### 4.2. Decryption Process

The previous example is used for the inverse operation (decryption). Figure (6) shows the QR codes for resulted QR, key & Message.

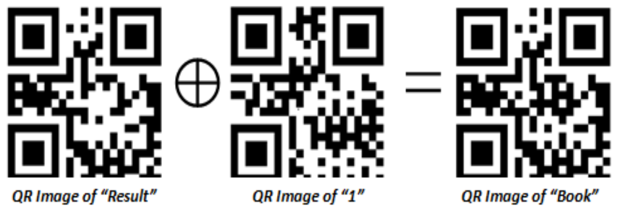


**Figure 6.** Decryption process

## 5. Conclusions

This method could be used in large scope. Since QR codes could be used for converting information to 2D barcode (QR code), this method can be used to encrypt any type of messages or files (numeric, URLs, alphanumeric and byte/binary) and send it to the receiver safely. Also, the method enables the user to store important data or information safely as QR. The information could be retrieved easily from the QR code using QR reader.

## REFERENCES

[1] P. Kieseberg, M. Leithner, M. Mulazzani, L. Munroe, S. Schrittwieser, M. Sinha, et al., "QR code security," in Proceedings of the 8th International Conference on Advances in Mobile Computing and Multimedia, 2010, pp. 430-435.

[2] S. Dey, S. Agarwal, and A. Nath, "Confidential Encrypted Data Hiding and Retrieval Using QR Authentication System," in Communication Systems and Network Technologies (CSNT), 2013 International Conference on, 2013, pp. 512-517.

[3] S. Dey, "SD-EQR: A New Technique To Use QR CodesTM in Cryptography," arXiv preprint arXiv:1205.4829, 2012.

[4] D. Chatterjee, J. Nath, S. Dasgupta, and A. Nath, "A new Symmetric key Cryptography Algorithm using extended MSA method: DJSA symmetric key algorithm," in Communication Systems and Network Technologies (CSNT), 2011 International Conference on, 2011, pp. 89-94.

[5] D. Sonawane, M. Upadhye, P. Bhogade, and S. Bajpai, "QR based Advanced authentication for all hardware platforms," International Journal of Scientific and Research Publications, vol. 4, pp. 1-4, 2014.

[6] M. Bajpai and A. P. Agrawal, "INTEGRATION OF 2D SECURE BARCODE IN IDENTITY CARDS: WITH ADDITIONAL SECURITY FEATURES."

[7] O. Sharaby, "Form Meets Function: Extreme Makeover QR Code Edition," ed: Archived from the original on, 2012.

[8] H. Chan, "How to: Make your QR codes more beautiful," Maskable, April, vol. 18, 2011.

[9] R. Cox. (2012). QArt Codes. Available: http://web.archive.org/web/20150321031237/http://research.swtch.com/qart.

[10] S. Hore, T. Bhattacharya, and S. B. Chaudhuri, "A Robust Medical Image Authentication Technique using QR Code and DWT," International Journal of Computer Applications, vol. 83, 2013.