A Family of Block Ciphers Based on Multiple Quasigroups

Umesh Kumar[1*] and V. Ch. Venkaiah[1]

[1*]School of Computer & Information Sciences, University of Hyderabad, Prof. C.R. Rao Road, P.O. Central University, Gachibowli, Hyderabad, 500046, Telangana, India.

*Corresponding author(s). E-mail(s): kumar.umesh285@gmail.com;
Contributing authors: vvcs@uohyd.ernet.in;

Abstract

A family of block ciphers parametrized by an optimal quasigroup is proposed in this paper. The proposed cipher uses sixteen $\mathbf{4 \times 4}$ bits S-boxes as an optimal quasigroup of order **16**. Since a maximum of **16!** optimal quasigroups of order **16** can be formed, the family consists of $\boldsymbol{C_1^{16!}}$ cryptosystems. All the sixteen S-boxes have the highest algebraic degree and are optimal with the lowest linearity and differential characteristics. Therefore, these S-boxes are secure against linear and differential attacks. The proposed cipher is analyzed against various attacks, including linear and differential attacks and we found it to be resistant to these attacks. The proposed cipher is implemented in C++, compared its performance with existing quasigroup based block ciphers, and we found that our proposal is more efficient than existing quasigroup based proposals. We also evaluated our cipher using various statistical tests of the NIST-STS test suite, and we found it to pass these tests. We also established in this study that the randomness of our cipher is almost the same as that of the AES-128.

Keywords: Block cipher, Cryptography, Latin square, NIST-STS, Optimal quasigroup, Quasigroup

MSC Classification: 11T71 , 94A60 , 68P25

1 Introduction

Finding new methods for creating cryptographic algorithms is one of the current trends in cryptography. Nowadays, a mathematical object called a quasigroup [27], popularly known as Latin square [9], has received lots of attention. It is a non-associative algebraic structure, widely used in the design of various cryptographic algorithms such as hash functions [7, 16], message authentication codes [16], secret sharing schemes [1, 29], stream ciphers [17, 28], block ciphers [4, 5, 30], and in many more applications [7, 28]. This is because the number of quasigroups grows exponentially with its order [26]. Also, quasigroup based cryptosystems are suitable for low-resource devices such as smartphones, sensors, tablets, etc [4]. This is in contrast to the most popular cryptosystems, such as AES, DES, and RSA, which drain the battery of such devices [5, 30].

This paper proposes a new cipher algorithm for the encryption/decryption of messages. It is a family of encryption systems parametrized by an optimal quasigroup. The family consists of $C_1^{16!}$ encryption systems since there are $C_1^{16!}$ ways to select an optimal quasigroup out of 16! optimal quasigroups. Once an encryption system is fixed, the set of 16 optimal quasigroups of order 16 it uses is fixed. These 16 optimal quasigroups are generated based on an original optimal quasigroup used. Also, these optimal quasigroups are constructed based on the 16 optimal S-boxes of 4×4 bits. For encrypting/decrypting a message of 128 bits, the proposed cipher performs a total of 17 rounds, and each round uses 128 bits round key along with one non-linear transformation and two linear transformations. The non-linear transformation is nothing but a quasigroup operation based on a key-dependent S-box. That is, for each quasigroup operation, non-linear transformation depends on the round key and chooses one S-box out of the 16 S-boxes. We believe that key-dependent S-box ciphers are more secure than fixed S-box ciphers. This is because key-dependent S-boxes do not offer any specific properties to the attackers. Most key-dependent S-box ciphers are effectively random. Examples of such ciphers are Blowfish [24] and SEAL [8]. The security of the proposed cipher is analyzed, and the randomness of the obtained ciphertext is tested. We noted that the proposed cipher satisfied all the required properties.

The paper is organized as follows: Next section gives a brief overview of quasigroups and optimal quasigroups, including quasigroup operations for encryption and decryption. Section 3 discusses related work. The building elements of the proposed cipher, including the generation of the round key and the related details, such as the avalanche effect of the expanded key and generation of the quasigroups, are presented in Section 4. The performance of the proposed cipher's algorithm and its comparison with the existing ciphers are discussed in Section 5. The security analysis of the proposed cipher is discussed in Section 6. The conclusion and future work are given in Section 7.

2 Mathematical background

2.1 Quasigroup

Definition 2.1. Let $\mathbb{Z}_n$ be the set of non-negative integers less than n. A quasigroup $Q = (\mathbb{Z}_n, *)$ defined over the set $\mathbb{Z}_n$ with a binary operation $*$ satisfies the following properties:

(i) For all $t_1, t_2 \in \mathbb{Z}_n$, $t_1 * t_2 \in \mathbb{Z}_n$, (Closure property).
(ii) For each pair $(t_1, t_2) \in \mathbb{Z}_n \times \mathbb{Z}_n$, there exists unique pair $(t_3, t_4) \in \mathbb{Z}_n \times \mathbb{Z}_n$, such that $t_1 * t_3 = t_2$ and $t_4 * t_1 = t_2$.

Quasigroups of order n are usually represented by an $n \times n$ multiplication table. Also, each row and each column of the multiplication table consists of the permutations of the elements of the set $\mathbb{Z}_n = \{0, 1, 2, \ldots, n-1\}$ in such a way that each element occurs exactly once in each row and exactly once in each column. Table 1 is an example of a quasigroup of order 16. Such a table is also called a Latin square [9]. Various algorithms exist in the literature [15, 21] that generate quasigroups of arbitrary order. Since the number of quasigroups grows exponentially with its order, the generation of all the quasigroups is a hard problem. An estimate on the number of quasigroups of order n is given by the following inequality [19]

$$\frac{(n!)^{2n}}{n^{n^2}} \leq Q(n) \leq \prod_{k=1}^{n} (k!)^{\frac{n}{k}}, \tag{1}$$

where $Q(n)$ denotes the number of quasigroups of order n.

2.2 Optimal quasigroup

Definition 2.2. An optimal quasigroup $Q = (\mathbb{Z}_{2^m}, *)$ is a groupoid which has the following properties:

(i) Q must be a quasigroup (see Definition 2.1).
(ii) Each row or each column of Q must be an optimal S-box of $m \times m$ bits.

An optimal quasigroup of order 2^m can be viewed as a collection of $m \times m$ bits optimal S-boxes. An $m \times m$ bits S-box is a Boolean map $S : F_2^m \to F_2^m$, where F_2 is Galois field of characteristic 2. In other words, an S-box is a permutation of the elements of $\mathbb{Z}_{2^m} = \{0, 1, 2, \ldots, 2^m - 1\}$. Our proposed cipher uses 4×4 bits optimal S-boxes to form an optimal quasigroup. Description of a 4×4 bits optimal S-box is given in Definition 2.3.

Definition 2.3. A 4×4 bits S-box is said to be optimal if the following conditions are satisfied [18]:

(i) S is a bijection,

(ii) $\mathbb{L}(S) = 8$, and
(iii) $\mathbb{D}(S) = 4$,

where $\mathbb{L}(S)$ and $\mathbb{D}(S)$ are the linearity and the differential characteristic of the S-box, which are defined as follows: Let $u = (u_0, u_1, \ldots, u_{m-1})$ and $v = (v_0, v_1, \ldots, v_{m-1})$ be two binary vectors. The dot product of u and v can be written as

$$u.v = \sum_{i=0}^{m-1} u_i.v_i, \tag{2}$$

then

$$\mathbb{L}(S) = max\{|\mathbb{W}_S(u,v)| : u \in F_2^m, v \in F_2^m \, and \, v \neq 0\} \tag{3}$$

and

$$\mathbb{D}(S) = max\{|\triangle_S\,(u,v)| : u \in F_2^m, v \in F_2^m \, and \, u \neq 0\}, \tag{4}$$

where

$$\mathbb{W}_S(u,v) = \sum_{x \in F_2^m} (-1)^{u.x+v.S(x)}$$

and

$$\triangle_S\,(u,v) = \{x \in F_2^m : S(x \oplus u) \oplus S(x) = v\},$$

$\oplus$ denotes bitwise `XOR` (modulo 2) operation and $|\cdot|$ denotes the cardinality of a set. The linearity and the differential characteristic of an S-box measure the resistance against the linear and differential cryptanalysis attacks, respectively. The smaller the linearity and the differential characteristic of an S-box, the more secure against these attacks.

As of yet, no algorithm has been developed for generating the optimal quasigroups of order $2^m, m \geq 4$. So, generating an optimal quasigroup of order 2^m is still a longstanding open problem. In this paper, we described an optimal quasigroup of order 16 using 16 optimal S-boxes of 4×4. In other words, each row of an optimal quasigroup is an optimal S-box. So, firstly, we studied various algorithms that exist in the literature [22, 31] to generate 4×4 bits optimal S-boxes. And for generating the optimal S-boxes of 4×4 bits, we used the algorithm described in [22]. We noted that all such S-boxes are not suitable to form an optimal quasigroup. This is because an optimal quasigroup is a mathematical object and has certain properties that must be satisfied (see Definition-2). We have chosen 16 S-boxes, namely, $S_0, S_1, S_2, \ldots, S_{15}$ given in Table 1. These 16 S-boxes are suitable to form an optimal quasigroup, $Q = (\mathbb{Z}_{16}, *)$, where $*$ is a quasigroup operation corresponding to the quasigroup Q of order 16.

Table 1 Optimal quasigroup of order 16.

	$*$	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**
S_0 :	**0**	10	0	1	12	3	13	4	15	14	9	6	7	5	8	2	11
S_1 :	**1**	11	2	8	5	7	6	9	14	15	4	13	3	12	1	0	10
S_2 :	**2**	5	8	2	11	14	9	6	7	3	13	4	15	10	0	1	12
S_3 :	**3**	8	5	11	2	9	14	7	6	13	3	15	4	0	10	12	1
S_4 :	**4**	9	14	7	6	8	5	11	2	0	10	12	1	13	3	15	4
S_5 :	**5**	2	11	5	8	6	7	14	9	4	15	3	13	1	12	10	0
S_6 :	**6**	12	1	0	10	15	4	13	3	7	6	9	14	11	2	8	5
S_7 :	**7**	3	13	4	15	10	0	1	12	5	8	2	11	14	9	6	7
S_8 :	**8**	14	9	6	7	5	8	2	11	10	0	1	12	3	13	4	17
S_9 :	**9**	7	6	9	14	11	2	8	5	12	1	0	10	15	4	13	3
S_{10} :	**10**	1	12	10	0	4	15	3	13	6	7	14	9	2	11	5	8
S_{11} :	**11**	6	7	14	9	2	11	5	8	1	12	10	0	4	15	3	13
S_{12} :	**12**	0	10	12	1	13	3	15	4	9	14	7	6	8	5	11	2
S_{13} :	**13**	4	15	3	13	1	12	10	0	2	11	5	8	6	7	14	9
S_{14} :	**14**	13	3	15	4	0	10	12	1	8	5	11	2	9	14	7	6
S_{15} :	**15**	15	4	13	3	12	1	0	10	11	2	8	5	7	6	9	14

2.3 Quasigroup operation for encryption and decryption

Sixteen optimal quasigroups of order 16 are employed in the design of the proposed block cipher. Therefore, all operations are performed in the form of 4-bit (also called nibbles) aggregations. Let each byte of data be divided into two nibbles. That is, a byte value x is represented as $x = x_1x_0$, where x_1 and x_0 are nibbles. Then, the quasigroup operations for encryption and decryption are defined as

$$x_1x_0 \star_i y_1y_0 = (x_1 *_i y_1)||(x_0 *_i y_0) \tag{5}$$

and

$$x_1x_0 \sharp_i y_1y_0 = (x_1 \setminus_i y_1)||(x_0 \setminus_i y_0), \tag{6}$$

respectively, where $(*_i, \setminus_i)$ and $(\star_i, \sharp_i)$ are quasigroup operations corresponding to nibbles and bytes respectively, $0 \leq i \leq 15$, and $||$ is a concatenation operation that concatenates two 4-bit values into one 8-bit value. Also, the symbol $\sharp_i$ is called the inverse (or left inverse) quasigroup operation concerning to the symbol $\star_i$. That is, if a quasigroup $Q_i = (\mathbb{Z}_n, \star_i)$ is used for encryption then its left inverse quasigroup $LIQ_i = (\mathbb{Z}_n, \sharp_i)$ is used for decryption. To find a left inverse quasigroup LIQ of a given quasigroup Q is defined in [17].

3 Related work

Various quasigroup based block ciphers exist in the literature [4, 5, 30]. These ciphers are designed based on $\{e, d\}$-transformations [7], and perform 32 rounds to encrypt or decrypt a block of data. In [4, 5], authors proposed two block ciphers based on a randomly chosen quasigroup of order 256. Note that they employed the same encryption/decryption algorithms in both their proposals. The ciphers are resistant to quasigroup attack (exhaustive quasigroup search) due to a large number of quasigroups of its order. But, a randomly chosen quasigroup may not be optimal from a linear and differential point of view. Also, it may be a challenge to store it in small devices. We tested the software

performance of these ciphers by varying the input and noted that they take around 10 seconds to encrypt a $4MB$ of data.

Zhao and Xu used an optimal quasigroup of order 16 to encrypt or decrypt a block of 64 bits [30]. The cipher is analyzed only on the basis of the algebraic properties of the optimal quasigroup used and not on the basis of the overall structure of the cipher. We tested the software performance of the cipher by varying the input and observed that the cipher is slower than that of [4, 5]. The cipher took around 13 seconds to encrypt a $4MB$ of data. Also, the cipher does not exhibit a good diffusion effect.

4 Proposed block cipher

The notation employed in this section are given in Table 2. Here, we discuss

Table 2 Some important notations.

Notation	Meaning
`XOR` or $\oplus$	:bitwise addition modulo 2 operation
IV	:an initial value of 128 bits
N	:a total number of blocks to be encrypted/decrypted
K_0	:initial (0^{th}) round key or secret key
RK_r	:r^{th} round key for encryption, $0 \leq r \leq 16$
RK_t	:t^{th} round key for decryption, where $t = 16 - r$
$RK_r : w_p$	:p^{th} word of the r^{th} round key for encryption, $0 \leq r \leq 16, 0 \leq p \leq 67$
$RK_t : w_p$	:p^{th} word of the t^{th} round key for decryption, where $t = 16 - r$
W	:a total of 68 words of the round key
r & t	:round number for encryption & decryption respectively, where $0 \leq r, t \leq 16$
B & C	:block number for encryption & decryption respectively, where $0 \leq B, C \leq N - 1$

the proposed block cipher. It is an iterative cipher, and its design is based on the Substitution Permutation Network (SPN). It uses a 128 bits secret key and 16 optimal quasigroups of order 16. These 16 optimal quasigroups are generated based on an original optimal quasigroup of order 16. Note that there are 16! original optimal quasigroups of order 16. These 16! optimal quasigroups can be generated by permuting the rows of an optimal quasigroup. So, our cipher is a family of encryption systems parametrized by an original optimal quasigroup. It performs a total of 17 rounds to encrypt or decrypt a block of 128 bits. Each round consists of a sequence of transformations. These transformations are an intermix of substitutions and permutations. Each round of the proposed cipher uses a random optimal quasigroup out of 16 optimal quasigroups except in the initial/last round of encryption/decryption, and this optimal quasigroup is selected on the basis of the previous/next round key of encryption/decryption. Because of these properties, the relationship between the plaintext and the corresponding ciphertext is not transparent and provides a higher level of security. Also, the proposed cipher leverages the space of a single optimal quasigroup and employs 16 optimal quasigroups by generating them from a single optimal quasigroup. That is, the space required by 16 optimal quasigroups is reduced to that of a single optimal quasigroup. The

Theorem 4.1 is useful in proving the correctness of the proposed cipher. The workflow of encryption and decryption of the proposed block cipher is shown in Fig. 1. In the figure, $w_p, 0 \leq p \leq 67$, denotes a 32-bit word of the round key. The word representation of the round key is described in the next sub-

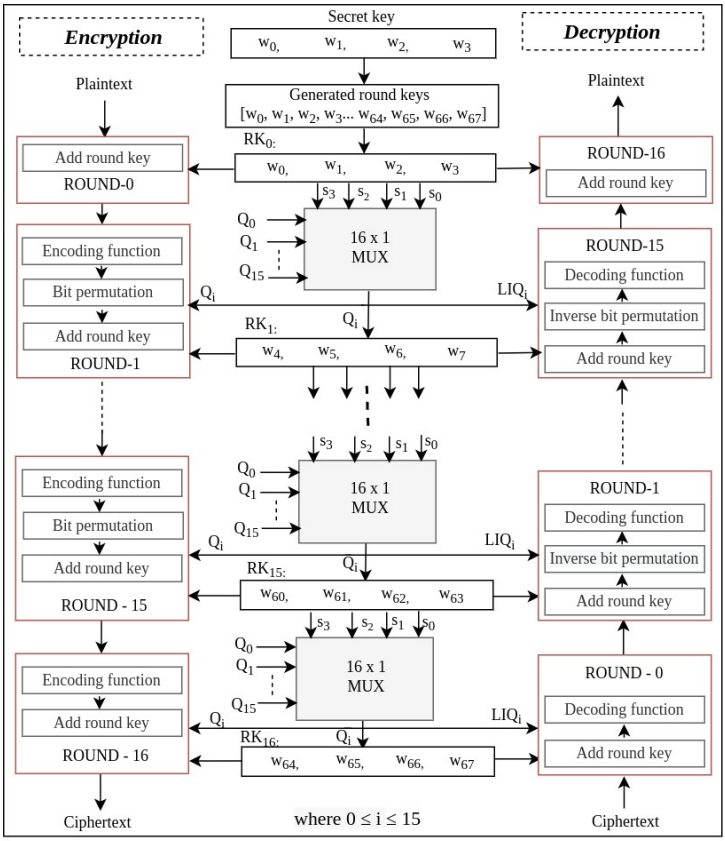

Fig. 1 Workflow of encryption and decryption of new block cipher.

section. Also, every round uses a quasigroup Q_i for encryption and an inverse quasigroup LIQ_i for decryption (where LIQ_i is the left inverse of Q_i), each of which is selected by a 16×1 multiplexer.

The algorithm of the proposed cipher consists of four parts: (1) an algorithm to randomly select an optimal quasigroup for each round of a block, (2) an algorithm to generate a round key, (3) an encryption algorithm, and (4) a decryption algorithm. The encryption algorithm employs three different transformations: (i) Encoding function, (ii) Bit permutation, and (iii) Add round key. Similarly, the decryption algorithm has three corresponding inverse transformations: (i) Decoding function, (ii) Inverse bit permutation, and (iii) Add round key.

4.1 Generation of round key

Our proposed block cipher uses a round key generation algorithm to encrypt or decrypt a block of data. It uses a 128-bit round key for each round to encrypt or decrypt a block of 128 bits. These round keys are generated based on the secret key of 128 bits along with 16 optimal quasigroups of order 16. The generation of each round key uses an optimal quasigroup out of 16 optimal quasigroups, selected by a 16×1 multiplexer (see subsection 4.2). Now, the secret key of 128 bits is partitioned into four words, and these, in turn, are arranged as four columns of a matrix. Let $K_0 = (k_{(0,0)}, k_{(0,1)}, k_{(0,2)}, k_{(0,3)}, \ldots, k_{(3,0)}, k_{(3,1)}, k_{(3,2)}, k_{(3,3)})$ be a secret key of 128 bits (16 bytes), where each $k_{(i,j)}$ is a byte value for $0 \leq i, j \leq 3$, which are organized as a 4×4 matrix of bytes as shown in Fig.2 (a). In this matrix, p^{th} word (column) is denoted by w_p, where $0 \leq p \leq 3$ and size of each w_p is 32 bits. These four words are used to create the initial (0^{th}) round key, and it is represented as $(K_0 : w_0, K_0 : w_1, K_0 : w_2, K_0 : w_3)$ or simply, $K_0 = (w_0, w_1, w_2, w_3)$. Our proposed cipher performs a total of 17 rounds for encrypting/decrypting a block of 128 bits, and each round consists of four words as a key. Therefore, a total of 68 words $(W = \{RK_0 : w_0, RK_0 : w_1, RK_0 : w_2, RK_0 : w_3, RK_1 : w_4, \ldots, RK_{15} : w_{63}, RK_{16} : w_{64}, RK_{16} : w_{65}, RK_{16} : w_{66}, RK_{16} : w_{67}\})$ are required as shown in Fig. 2. These 68 words are generated based on K_0, includ-

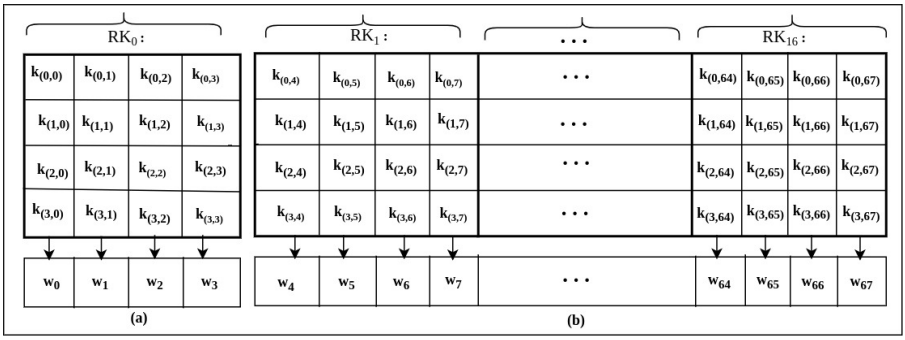

Fig. 2 Representation of round keys.

ing four words of the initial (0^{th}) round key K_0. For $1 \leq r \leq 16$, $4 \leq p \leq 67$, a p^{th} word of the r^{th} round key is generated based on the previous words $p-4$ and $p-3$. The algorithm of the round key generation is given in Algorithm 1. In this algorithm $Q_i, 0 \leq i \leq 15$, is a generated quasigroup. And $\bigstar_i$ is the binary operation (quasigroup operation) of 32-bit words defined as follows: Let $U = (u_1, u_2, u_3, u_4)$ and $V = (v_1, v_2, v_3, v_4)$ be two words of 32 bits each, where u_j and v_j are byte values $(1 \leq j \leq 4)$. Then,

$$U \bigstar_i V = Z = (u_1 \star_i v_1, u_2 \star_i v_2, u_3 \star_i v_3, u_4 \star_i v_4),$$

Algorithm 1 : Generation of the round key

Input: 1. A 128-bit secret key in the form of a 4×4 matrix of bytes as shown in Fig. 2(a).
2. An optimal quasigroup of order 16.

Output: Generates all the 17 rounds key in the form of a 4×68 matrix of bytes as shown in Fig. 2.

if $(r = 0$ and $0 \leq p \leq 3)$ **then**
  $RK_0 = K_0$
else
  for $(p = 4$ to $67)$ **do**
    if $(p \bmod 4 = 0)$ **then**
      Q_i = quasigroup generated based on r^{th} round key.
      $r = r + 1$
    end if
    if $(RK_{(r-1)} : w_{(p-3)} \in W)$ **then**
      $RK_r : w_p = RK_{(r-1)} : w_{(p-4)} \bigstar_i RK_{(r-1)} : w_{(p-3)}$
    else
      $RK_r : w_p = RK_{(r-1)} : w_{(p-4)} \bigstar_i RK_r : w_{(p-3)}$
    end if
  end for
end if

where $\star_i$ is one of the quasigroup operations defined in Eq. (5), $0 \leq i \leq 15$. Note that a quasigroup operation $\bigstar_i$ is nothing but a look-up table operation. So, here the resultant value Z is determined by looking up the element having the row number U and the column number V in the table representation of the quasigroup $Q = (\mathbb{Z}_{2^{32}}, \bigstar_i)$.

Note that both the encryption and the generation of the round key algorithms use the same set of optimal quasigroups. These optimal quasigroups are generated based on an original optimal quasigroup $Q = (\mathbb{Z}_{16}, *)$. But in decryption, we use a left inverse quasigroup $LIQ = (\mathbb{Z}_{16}, \backslash)$ of the quasigroup Q. Note that both the encryption and the decryption algorithms use the same round key. Therefore in decryption, the key generation algorithm has to perform the quasigroup operations corresponding to the quasigroup $Q = (\mathbb{Z}_{16}, *)$ based on the quasigroup $LIQ = (\mathbb{Z}_{16}, \backslash)$. The relation between these two quasigroup operations is as follows [17]:

$$p \backslash s = r \Leftrightarrow p * r = s, \forall (p, q, r) \in \mathbb{Z}_{16} \times \mathbb{Z}_{16} \times \mathbb{Z}_{16}.$$

4.2 Generation of quasigroups

Our proposed cipher uses 16 optimal quasigroups of order 16 for either encrypting or decrypting a block of data. Of these, only one optimal quasigroup is used in each round, and an optimal quasigroup may be used in more than one round. This is decided by a 16×1 multiplexer used in each round of the

proposed cipher as shown in Fig. 1. A multiplexer is a combinational circuit that has a maximum of 2^n input values for n selection lines and it produces a single output. In our case n is 4. The multiplexer along with its truth table are shown in Fig. 3. This multiplexer selects an optimal quasigroup for each

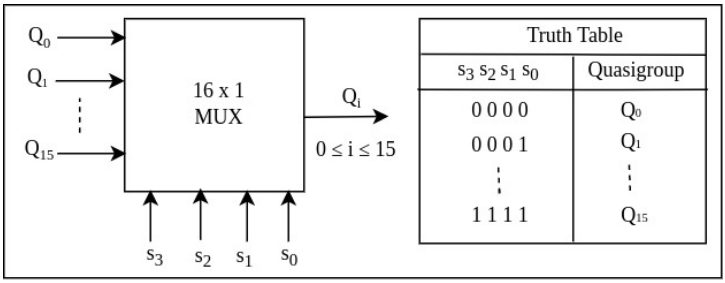

Fig. 3 16×1 multiplexer and its truth table.

round based on the current state of selection lines s_3, s_2, s_1, s_0. The value of $s_\alpha (0 \leq \alpha \leq 3)$ is either 0 or 1, which is updated by both the round key and an optimal quasigroup of the previous round of the proposed encryption system, except the 1^{st} round. This is because the 1^{st} round uses the initial (0^{th}) round key and an original optimal quasigroup Q such as the one given in Table 1. If $s_3 = 0, s_2 = 0, s_1 = 0, s_0 = 0$, then the multiplexer selects an optimal quasigroup Q_0. If $s_3 = 0, s_2 = 0, s_1 = 0, s_0 = 1$, then the multiplexer selects an optimal quasigroup Q_1, and so on. The values of s_3, s_2, s_1 and s_0 are updated as follows: For selecting $(r+1)^{th}$ round optimal quasigroup, we consider r^{th} (previous) round key shown in Fig. 2. This is a 4×4 matrix of bytes. Now, define

$$
\begin{aligned}
\texttt{Temp}_0 &= k_{(0,4r)} \oplus k_{(1,4r)} \oplus k_{(2,4r)} \oplus k_{(3,4r)}, \\
\texttt{Temp}_1 &= k_{(0,4r+1)} \oplus k_{(1,4r+1)} \oplus k_{(2,4r+1)} \oplus k_{(3,4r+1)}, \\
\texttt{Temp}_2 &= k_{(0,4r+2)} \oplus k_{(1,4r+2)} \oplus k_{(2,4r+2)} \oplus k_{(3,4r+2)}, \\
\texttt{Temp}_3 &= k_{(0,4r+3)} \oplus k_{(1,4r+3)} \oplus k_{(2,4r+3)} \oplus k_{(3,4r+3)}, \\
\texttt{XORofTemp}_j &= \texttt{Temp}_0 \oplus \texttt{Temp}_1 \oplus \texttt{Temp}_2 \oplus \texttt{Temp}_3.
\end{aligned}
$$

Each $\texttt{Temp}_j$ for $0 \leq j \leq 3$ and $\texttt{XORofTemp}_j$ are byte values. Let $\texttt{XORofTemp}_j$ be divided into two 4-bit values (nibbles), that is $\texttt{XORofTemp}_j = x_1 x_0$, where x_0 and x_1 are nibbles. Then,

$$
\texttt{Rconstval} = x_1 *_i x_0, \tag{7}
$$

where $*_i$ is the quasigroup operation corresponding to the r^{th} round's optimal quasigroup, $0 \leq i, r \leq 15$. The Rconstval is 4 bits and they are denoted by s_3, s_2, s_1, and s_0, where s_0 and s_3 are the least significant bit and the

most significant bit, respectively. These s_α, $0 \leq \alpha \leq 3$, are considered as the selection lines of the 16×1 multiplexer.

Each round of our proposed cipher uses an optimal quasigroup of order 16. One such optimal quasigroup Q is shown in Table 1. By permuting the rows of the optimal quasigroup Q, 16! optimal quasigroups can be created. That is, these 16! optimal quasigroups are nothing but permutations of $S_0, S_1, S_2, \ldots,$ and S_{15} S-boxes. These 16 S-boxes are shown in Table 1. Note that our cipher uses only 16 optimal quasigroups. So, we select any 16 out of the total 16! optimal quasigroups. Let the selected optimal quasigroups be denoted by $Q_0 = (\mathbb{Z}_{16}, *_0), Q_1 = (\mathbb{Z}_{16}, *_1), ..., Q_{15} = (\mathbb{Z}_{16}, *_{15})$, where $*_0, *_1, ..., *_{15}$ are the quasigroup operations corresponding to $Q_0, Q_1, ..., Q_{15}$, respectively. Note that all these 16 optimal quasigroups need not be stored. This is because each optimal quasigroup consists of the same S-boxes but in a different order (permutation). A permutation of the S-boxes is generated based on the previous round key in the case of encryption; whereas it is the next round key for the case of decryption. Let $Q = (S_0, S_1, S_2, \ldots, S_{15})$ be an original optimal quasigroup used for encryption. Then a new permutation of Q corresponding to an $(r+1)^{th}$ round optimal quasigroup is generated using the following equation:

$$Q = (S_{(0+\texttt{Rconstval})\ mod\ 16}, S_{(1+\texttt{Rconstval})\ mod\ 16}, \ldots, S_{(15+\texttt{Rconstval})\ mod\ 16}) \quad (8)$$

where `Rconstval` is a round constant value, which is determined by Eq. (7), $0 \leq$ `Rconstval` ≤ 15. Note that the resultant optimal quasigroup $Q \in \{Q_0, Q_1, \ldots, Q_{15}\}$. Also, the same permutations of the S-boxes are used for the decryption, but these permutations are generated based on the LIQ, where LIQ is the left inverse quasigroup of the original optimal quasigroup Q. The correctness of this is proven by Theorem 4.1.

Theorem 4.1 *Let Q and LIQ be respectively a quasigroup and its left inverse quasigroup. Let Q_x and LIQ_x be the result of applying a permutation $\mathcal{P}$ on the rows of Q and LIQ, respectively. Then, LIQ_x is the left inverse quasigroup of the quasigroup Q_x.*

Proof : Let $Q = (\mathbb{Z}_n, *)$ and $LIQ = (\mathbb{Z}_n, \backslash)$ be quasigroups of order n, whose operation tables look like the ones given in Table 3 (a) and Table 3 (b), respectively. Since LIQ is the left inverse quasigroup of Q, we have

$$i * j = a_{i,j} \Leftrightarrow i \backslash a_{i,j} = b_{i,j}$$
$$\forall\ i, j \in \{0, 1, \ldots, n-1\}.$$

Now let

$$\mathcal{P} = \begin{pmatrix} \mathcal{P}(0) & \mathcal{P}(1) & \mathcal{P}(2) & \mathcal{P}(2) & \cdots & \mathcal{P}(n-1) \\ 0 & 1 & 2 & 3 & \cdots & n-1 \end{pmatrix}$$

be a permutation applied on the rows of the quasigroups Q and LIQ, and let $Q_x = (\mathbb{Z}_n, *_x)$ and $LIQx = (\mathbb{Z}_n, \backslash_x)$ denote the resulting quasigroups, where $\mathcal{P}(i)^{th}$ rows of both the Q and the LIQ becomes the i^{th} rows of both the Q_x and the LIQ_x,

respectively, $0 \leq i, \mathcal{P}(i) \leq n-1$. That is, the operation tables of Q_x and LIQ_x will be as shown in Table 4 (a) and Table 4 (b), respectively. Therefore,

$$i *_x j = a_{\mathcal{P}(i),j} \Leftrightarrow i \setminus_x a_{\mathcal{P}(i),j} = b_{\mathcal{P}(i),j}$$
$$\forall\, i, j \in \{0, 1, \ldots, n-1\}.$$

This is true because

$$i * j = a_{i,j} \Leftrightarrow i \setminus a_{i,j} = b_{i,j}$$
$$\forall\, i, j \in \{0, 1, \ldots, n-1\}.$$

□

Table 3 Representation of quasigroups Q and LIQ of order n.

$*$	**0**	**1**	...	$n-1$
0	$a_{0,0}$	$a_{0,1}$	...	$a_{0,n-1}$
1	$a_{1,0}$	$a_{1,1}$	...	$a_{1,n-1}$
2	$a_{2,0}$	$a_{2,1}$	...	$a_{2,n-1}$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
$n-1$	$a_{n-1,0}$	$a_{n-1,1}$	...	$a_{n-1,n-1}$

(a)

$\setminus$	**0**	**1**	...	$n-1$
0	$b_{0,0}$	$b_{0,1}$	...	$b_{0,n-1}$
1	$b_{1,0}$	$b_{1,1}$	...	$b_{1,n-1}$
2	$b_{2,0}$	$b_{2,1}$	...	$b_{2,n-1}$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
$n-1$	$b_{n-1,0}$	$b_{n-1,1}$	...	$b_{n-1,n-1}$

(b)

Table 4 Representation of quasigroups Q_x and LIQ_x of order n.

$*_x$	**0**	**1**	...	$n-1$
0	$a_{\mathcal{P}(0),0}$	$a_{\mathcal{P}(0),1}$	...	$a_{\mathcal{P}(0),n-1}$
1	$a_{\mathcal{P}(1),0}$	$a_{\mathcal{P}(1),1}$	...	$a_{\mathcal{P}(1),n-1}$
2	$a_{\mathcal{P}(2),0}$	$a_{\mathcal{P}(2),1}$	...	$a_{\mathcal{P}(2),n-1}$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
$n-1$	$a_{\mathcal{P}(n-1),0}$	$a_{\mathcal{P}(n-1),1}$	...	$a_{\mathcal{P}(n-1),n-1}$

(a)

$\setminus_x$	**0**	**1**	...	$n-1$
0	$b_{\mathcal{P}(0),0}$	$b_{\mathcal{P}(0),1}$	...	$b_{\mathcal{P}(0),n-1}$
1	$b_{\mathcal{P}(1),0}$	$b_{\mathcal{P}(1),1}$	...	$b_{\mathcal{P}(1),n-1}$
2	$b_{\mathcal{P}(2),0}$	$b_{\mathcal{P}(2),1}$	...	$b_{\mathcal{P}(2),n-1}$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
$n-1$	$b_{\mathcal{P}(n-1),0}$	$b_{\mathcal{P}(n-1),1}$	...	$b_{\mathcal{P}(n-1),n-1}$

(b)

The application of Theorem 4.1 is illustrated in Example 4.2.

Example 4.2. Let $Q = (\mathbb{Z}_8, *)$ be a quasigroup over $\mathbb{Z}_8 = \{0, 1, 2, 3, 4, 5, 6, 7\}$. Its left inverse quasigroup is $LIQ = (\mathbb{Z}_8, \setminus)$. These are shown in Table 5 (a) and Table 5 (b), where $*$ and $\setminus$ are the quasigroup operations corresponding to the quasigroups Q and LIQ, respectively. Let

$$\mathcal{P} = \begin{pmatrix} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 \\ 1 & 0 & 3 & 2 & 5 & 4 & 7 & 6 \end{pmatrix}$$

be a permutation applied on the rows of the quasigroup Q, and let $Q_x = (\mathbb{Z}_8, *_x)$ denote the resulting quasigroup. Then Q_x is as shown in Table 6 (a), where $*_x$ is the quasigroup operation corresponding to the quasigroups Q_x. Now, we applied the same rows permutation $\mathcal{P}$ on the left inverse quasigroup LIQ, and let $LIQ_x = (\mathbb{Z}_8, \setminus_x)$ denote the resulting quasigroup. Then LIQ_x

is as shown in Table 6 (b), where $\backslash_x$ is the left inverse quasigroup operation corresponding to the quasigroup LIQ_x. Using Theorem 4.1, it can be verified that the quasigroup LIQ_x is the left inverse quasigroup of the quasigroup Q_x.

Table 5 Quasigroups Q and LIQ of order 8.

$*$	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
0	5	1	6	3	4	0	2	7
1	1	6	3	4	0	2	7	5
2	6	3	4	0	2	7	5	1
3	3	4	0	2	7	5	1	6
4	4	0	2	7	5	1	6	3
5	0	2	7	5	1	6	3	4
6	2	7	5	1	6	3	4	0
7	7	5	1	6	3	4	0	2

(a)

$\backslash$	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
0	5	1	6	3	4	0	2	7
1	4	0	5	2	3	7	1	6
2	3	7	4	1	2	6	0	5
3	2	6	3	0	1	5	7	4
4	1	5	2	7	0	4	6	3
5	0	4	1	6	7	3	5	2
6	7	3	0	5	6	2	4	1
7	6	2	7	4	5	1	3	0

(b)

Table 6 Quasigroups Q_x and LIQ_x of order 8.

$*_x$	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
0	1	6	3	4	0	2	7	5
1	5	1	6	3	4	0	2	7
2	3	4	0	2	7	5	1	6
3	6	3	4	0	2	7	5	1
4	0	2	7	5	1	6	3	4
5	4	0	2	7	5	1	6	3
6	7	5	1	6	3	4	0	2
7	2	7	5	1	6	3	4	0

(a)

$\backslash_x$	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
0	4	0	5	2	3	7	1	6
1	5	1	6	3	4	0	2	7
2	2	6	3	0	1	5	7	4
3	3	7	4	1	2	6	0	5
4	0	4	1	6	7	3	5	2
5	1	5	2	7	0	4	6	3
6	6	2	7	4	5	1	3	0
7	7	3	0	5	6	2	4	1

(b)

4.3 Encryption

Encryption is carried out in a total of 17 rounds. Each of these rounds, except the initial and the last rounds, comprises the following three transformations and encrypts a 128-bit block of data. Initial (0^{th}) round performs only the add roun key, and the last (16^{th}) round performs encoding function and add round key.

4.3.1 Encoding function

The encoding function (denoted by $f_{(\star_i,K)}$) is one of the non-linear transformations of the proposed cipher. It uses 16 S-boxes in the form of an optimal quasigroup, given in Table 1. It is nothing but a key-dependent S-box layer that substitutes a byte for a byte, where the selection of the S-boxes depends on the round key. In other words, it is a substitution (S-box) layer of 16×8 bits, which is implemented with two S-boxes of 8×4 bits. It adds to the confusion property and hides the relationship between the key and the ciphertext; thereby

making it difficult to find the key from the ciphertext. Let $B=\{p_0, p_1, \ldots, p_{15}\}$, $K=\{k_0, k_1, \ldots, k_{15}\}$, and $C = \{c_0, c_1, \ldots, c_{15}\}$ denote input to a round, round key, and the output of a round, respectively. Then the way of using the encoding function $f_{(\star_i, K)}$ on B with round key K to obtain the corresponding C is as follows:

$$c_j = k_j \star_i p_j \tag{9}$$

where all p_j, c_j and k_j are byte values for $0 \leq j \leq 15$ and $\star_i$ is one of the quasigroup operations defined in Eq. (5), for $0 \leq i \leq 15$.

4.3.2 Bit permutation

This is a linear transformation used to increase the diffusion power of the cipher. This is the second transformation in a round function. This layer spreads the non-zero bits that increase the number of active S-boxes in a linear or differential trail, and we get the maximum impact of the substitution layer. It has the ability to hide the relationship between the plaintext and the ciphertext and hence aids in increasing diffusion. It maps the y^{th} bit to $P(y)^{th}$bits, where

$$P(y) = \left(1 + 32\left(\left(3\left\lfloor\frac{y \bmod 16}{4}\right\rfloor + (y \bmod 4)\right) \bmod 4\right) + 4\left\lfloor\frac{y}{16}\right\rfloor + (y \bmod 4)\right) \bmod 128 \tag{10}$$

By examining the permutation layer one can make the following observations:

1. The four output bits of a particular round S-box enter into four distinct S-boxes of the next round.
2. The four input bits to an S-box of a particular round come from four distinct S-boxes of the previous round.
3. The i^{th} bit output of an S-box of a particular round becomes the $((i + 1) \bmod 4)^{th}$ bit input of a distinct S-box of the next round, where $0 \leq i \leq 3$, 0 and 3 are the least and the most significant bits of the S-box.
4. The i^{th} bit input to an S-box of a particular round comes from the $((i - 1) \bmod 4)^{th}$ bit output of a distinct S-box of the previous round.

Based on the (1) observation; the four bits output of an S-box in one round will affect four S-boxes in the next round, and then 16 S-boxes together in the round after that. Therefore, it can be demonstrated that this permutation will affect all the 32 S-boxes in three rounds using similar reasoning for the subsequent rounds. That is, this bit permutation needs four rounds to achieve the full diffusion, that is, an input bit to an S-box of a particular round will influence all the 128 bits in four rounds which is optimal [2].

4.3.3 Add round key

This is also a linear transformation and it is the third and the last transformation in a round function. This transforms a round input $B=\{p_0, p_1, \ldots, p_{15}\}$ of 16 bytes to the corresponding round output $C = \{c_0, c_1, \ldots, c_{15}\}$ of 16 bytes by XORing the input B with a round key $K=\{k_0, k_1, \ldots, k_{15}\}$ of 16 bytes as follows:

$$c_j = k_j \oplus p_j \tag{11}$$

where all p_j, c_j and k_j are byte values for $0 \leq j \leq 15$.

4.3.4 Encryption algorithm based on CBC mode of operation

The algorithm of the proposed cipher is implemented using the Cipher Block Chaining (CBC) mode of operation. Each iteration of the proposed cipher encrypts/decrypts 128 bits of plaintext/ciphertext, and it is repeated until the entire plaintext/ciphertext is encrypted/decrypted. It can also be described using other modes of operation such as Cipher Feedback (CFB) mode , Output Feedback (OFB) and Counter (CTR) mode. Each mode of operation has its own advantages and disadvantages [25]. The encryption algorithm of the proposed block cipher is given in Algorithm 2. In this algorithm, Q_j, $0 \leq j \leq 15$,

Algorithm 2 : Encryption algorithm

Input: 1. Plaintext in the form of 128-bit (16 bytes) blocks. Let $B_0, B_1, \ldots, B_{(N-1)}$ be the N number of blocks to be encrypted.
2. Initial value (IV) of 128 bits (16 bytes).
3. Secret key of 128 bits (16 bytes).
4. An optimal quasigroup of order 16.

Output: Ciphertext whose size is equal to the size of the plaintext.

1: $\{RK_r{:}0 \leq r \leq 16\}$=Generate all the 17 rounds key.
2: **for** ($i = 0$ to $N-1$) **do**
3: B_i =XOR(B_i, IV)
4: B_i = Add round key(B_i, RK_0)
5: **for** ($r = 1$ to 15) **do**
6: Q_j = Generated quasigroup based on RK_{r-1}
7: B_i = Encoding function(B_i, RK_r, Q_j)
8: B_i = Bit permutation(B_i)
9: B_i = Add round key(B_i, RK_r)
10: **end for**
11: Q_j = Generated quasigroup based on RK_{15}
12: B_i = Encoding function(B_i, RK_{16}, Q_j)
13: B_i = Add round key(B_i, RK_{16})
14: $IV = B_i$
15: **end for**

is a generated quasigroup based on the $(r-1)^{th}$ round key for the r^{th} round of B_i block. The encryption algorithm updates block B_i many times during the encryption and the results are stored in the same block B_i, for $0 \leq i \leq N-1$.

4.4 Decryption

The decryption process is the reverse of encryption. It obtains the corresponding plaintext from the ciphertext. The algorithm of decryption is not the same as encryption. It uses the same sequence of round keys but in reverse order.

Also, it uses the inverses of the quasigroups that were used in the encryption. It also performs 17 rounds to decrypt a 128-bit block of data. Each round, except the initial and the last rounds, comprises of the following three transformations. In the initial round, add round key and decoding function; in the last round, only the add round key takes place.

4.4.1 Add round key

The add round key transformation for decryption is the same as that of the encryption process as described in sub-section 4.3.3.

4.4.2 Inverse bit permutation

The inverse bit permutation is the reverse of the bit permutation as defined by the Eq. (10), that is, the $P(y)^{th}$ bit maps to y^{th} bit.

4.4.3 Decoding function

The decoding function (denoted by $f_{(\sharp_i,K)}$) is the inverse of the encoding function. It uses the inverse of the same optimal quasigroup that was used in the encoding function. Let $C = \{c_0, c_1, \ldots, c_{15}\}$, $K=\{k_0, k_1, \ldots, k_{15}\}$, and $B=\{p_0, p_1, \ldots, p_{15}\}$ denote round input, round key, and the round output, respectively. Then the decoding function $f_{(\sharp_i,K)}$ on C with the round key K to recover B is as follows:

$$p_j = k_j \sharp_i c_j \tag{12}$$

where all p_j, c_j and k_j are byte values for $0 \leq j \leq 15$ and $\sharp_i$ is one of the left inverse quasigroup operations defined in Eq. (6).

4.4.4 Decryption algorithm based on CBC mode

The decryption algorithm of our proposed cipher is also implemented using the Cipher Block Chaining (CBC) mode of operation. Each iteration of the decryption algorithm decrypts 128 bits of the ciphertext and is repeated until the whole ciphertext is decrypted. The algorithm of decryption algorithm is given in Algorithm 3. In this algorithm, LIQ_j, $0 \leq j \leq 15$, is a generated left inverse quasigroup based on the $(t+1)^{th}$ (next) round key for the t^{th} round of C_i block. The `CopyofC`$_i$ is a temporary variable used to store the value of C_i before starting the decryption process. The decryption algorithm updates block C_i many times during the decryption, and the results are stored in the same block C_i, for $0 \leq i \leq N-1$.

5 Implementation and Performance analysis

The proposed cipher has been implemented in C++ on a system with the following configuration: Intel(R) Core(TM) i5-2400 CPU @3.40 GHz processor with 8 GB RAM and 64-bit Linux operating system. The source code of the proposed cipher is run 1000 times for different samples and we calculated the

Algorithm 3 : Decryption algorithm

Input: 1. Ciphertext in the form of 128-bit (16 bytes) blocks. Let $C_0, C_1, \ldots, C_{(N-1)}$ be the N number of blocks to be decrypted.
2. Initial value (IV) of 128 bits (16 bytes).
3. Secret key of 128 bits (16 bytes).
4. A left inverse quasigroup of order 16, this quasigroup is the left inverse of a quasigroup that was used in the encryption algorithm.

Output: Plaintext whose size is equal to the size of the ciphertext.

1: $\{RK_t{:}0 \leq t \leq 16\}$=Generate all the 17 rounds key.
2: **for** ($i = 0$ to $N - 1$) **do**
3: `CopyOf`C_i= C_i
4: LIQ_j = Generated quasigroup based on RK_1
5: C_i = Add round key (C_i, RK_0)
6: C_i = Decoding function (C_i, RK_0, LIQ_j)
7: **for** ($t = 1$ to 15) **do**
8: LIQ_j = Generated quasigroup based on RK_{t+1}
9: C_i = Add round key (C_i, RK_t)
10: C_i = Inverse bit permutation (C_i)
11: C_i =Decoding function (C_i, RK_t, LIQ_j)
12: **end for**
13: C_i = Add round key (C_i, RK_{16})
14: C_i = `XOR` (C_i, IV)
15: IV = `CopyOf`C_i
16: **end for**

average execution time in seconds. We have used the C++ standard <`chrono`> library to measure the execution time [14]. The performance of the proposed cipher is compared with that of the existing quasigroup based block ciphers [4, 5, 30], DES and AES-128. The results of this analysis are shown in Table 7. Also given in the last column of the table is the space required to store S-boxes or quasigroups in bytes corresponding to the existing and the proposed ciphers. According to the results, as shown in Table 7, we can observe that our proposed cipher is slightly slower than that of AES-128, but our cipher uses half the space compared to AES-128. In addition, our cipher is more efficient than [4, 5, 30] and DES.

Table 7 Comparison of the average execution time.

	Execution time in seconds (speed)			Space complexity (in bytes)
Block ciphers	1 MB	2.11 MB	4.21 MB	
[4, 5]	2.51	4.61	10.57	65536
[30]	3.32	6.71	13.47	128
DES	10.42	20.79	39.82	180
AES-128	0.67	1.21	2.32	256
Proposed cipher	1.23	2.47	4.95	128

6 Security analysis

The proposed cipher is a family of encryption systems parameterized by an optimal quasigroup of order 16. The sender and the receiver agree on a cryptosystem by first deciding on an optimal quasigroup. Since there are $C_1^{16!}$ ways to select an optimal quasigroup, the family consists of $C_1^{16!}$ cryptosystems. Once a cryptosystem is fixed, the set of 16 optimal quasigroups of order 16 it uses is fixed, and each round uses only one of these 16 quasigroups with equal probability, depending on the secret key. To access these 16 quasigroups, a cryptanalyst must first determine the secret key to be used.

- *Exhaustive key search attack*:- The proposed cipher uses a secret key of 128 bits. Therefore, a number of the possible keys is $2^{128} \approx 3.4 \times 10^{38}$. So, the running time of this attack is $T = O(u) = O(2^{128})$, where u is the size of the key space, it is an exponential. Also, let us assume an attacker uses a supercomputer and tries 5.37×10^{17} keys per second, then the attacker needs around 2.01×10^{13} years to determine the employed key. This is because these days the supercomputers can perform 5.37×10^{17} FLOPS[1] [11].

6.1 Linear cryptanalysis

Linear cryptanalysis is a powerful attack against a block cipher. This attack model works based on the known-plaintext. It uses linear relations of the plaintext bits, ciphertext bits, and sub-keys that hold with a suitably high probability. We further call them linear approximations. Let P_L be the probability of a linear approximation, then its probability bias (denoted by ϵ) can be defined as $|P_L - \frac{1}{2}|$. The higher the magnitude of the probability bias ϵ, the fewer known plaintexts are required to mount a linear attack. The proposed block cipher consists of 17 rounds, in which only the 16 rounds (from the 1^{st} round to the 16^{th} round of the encryption algorithm) use the substitutions (or encoding function). And each round uses an optimal quasigroup of order 16. That is, each round uses the 16 optimal S-boxes in the form of an optimal quasigroup.

There are various ways to mount the linear attack. In this paper, we have followed the procedure described in [13]. We constructed the linear approximations upto 15 rounds (that is, from the 1^{st} round upto the 15^{th} round). This is because, once a 15-round linear approximation is discovered with a suitably high linear probability bias ϵ, then it is conceivable to attack the cipher by recovering bits from the last (16^{th}) round key of the cipher [13]. For u bits of the plaintext or input to the 0^{th} round of the proposed cipher (denoted by $\mathcal{I}_0$), for v bits of the output of the 15^{th} round of the proposed cipher (denoted by $\mathcal{O}_{15}$), and a total of w bits of the round keys used from the 0^{th} round to the 15^{th} round of the proposed cipher (denoted by $\mathcal{K}_{(0,15)}$), such an approximation

[1] floating point operations per second

is defined as follows

$$\left(\bigoplus_{i=0}^{u} \mathcal{I}_0^{x_i}\right) \oplus \left(\bigoplus_{j=0}^{v} \mathcal{O}_{15}^{x_j}\right) = \bigoplus_{k=0}^{w} \mathcal{K}_{(0,15)}^{x_k} \tag{13}$$

where x_i, x_j and x_k denote bit positions, and $\oplus$ is a bitwise addition modulo 2 operation. The value of the right side of Eq. (13) would be either 0 or 1 depending on the round key bits involved. These bits are fixed but unknown (as they are determined by the key under attack). This kind of linear relation is obtained by concatenating the appropriate linear approximations of S-boxes from round to round of the cipher. This is because S-boxes are the only non-linear components of the proposed cipher. These linear approximations hold input and output bits of the S-boxes with certain probability.

We investigate the construction of a linear trail by examining non-zero inputs corresponding to the S-boxes from the 1^{st} round to the 15^{th} round. Using the LATs of the S-boxes in each round, if a particular non-zero input occurs then the corresponding output is decided with suitably high probability bias and least hamming weight. It is observed that the LATs of all the 16 S-boxes have the same magnitude linear probability bias values ($|P_L - \frac{1}{2}|$). Because of this, the key-dependent non-linear (S-box) layer called encoding function does not need to differentiate among LATs of various S-boxes while arriving at the optimal linear trail. We only consider the linear approximations of the S-boxes that have non-zero inputs and hence non-zero outputs to estimate the number of active S-boxes of a linear trail. Since the size of the input block of the proposed cipher is 128 bits, it is impossible to find all of the possible linear trails for 2^{128} inputs. So, we divide 128 bits input block into 4 sub-blocks as X_1, X_2, X_3, and X_4, where the size of each sub-block is 32 bits. And then, we use all the possible non-zero inputs of the first sub-block X_1 keeping all the remaining sub-blocks X_2, X_3, and X_4 as zero values. This gives rise to a maximum of 2^{32} linear trails. Similarly, we repeated the same procedure for the sub-blocks X_2, X_3, and X_4. We used a computer-based search to find an optimal linear trail by evaluating the number of active S-boxes at each round of the proposed cipher, and we found that the minimum number of active S-boxes in an optimal linear trail is 190. Also, the maximum probability bias of the S-boxes of the proposed cipher is 2^{-2}. Therefore, using the piling-up lemma we obtain that the probability bias ϵ for r rounds (here, $r = 15$) is as follows [20]:

$$\begin{aligned}\epsilon &= 2^{\#-1} \times (\text{Max. probability bias of an S-box})^{\#} \\ &= 2^{189} \times (2^{-2})^{190} \\ &= 2^{-191},\end{aligned}$$

where # denotes the minimum number of active S-boxes in an optimal linear trail. Hence, for 15 rounds the maximum probability bias of the proposed cipher is 2^{-191}. The complexity of the linear cryptanalysis attack can be

calculated by using the following formula [13]:

$$N_L \approx \frac{1}{\epsilon^2} = \frac{1}{(2^{-191})^2} = 2^{382},$$

where N_L denotes the number of known plaintexts. Hence the required number of known plaintexts is 2^{382} which is much greater than 2^{128}. This shows that the proposed cipher is resistant to linear cryptanalysis attacks.

6.2 Differential cryptanalysis

Differential cryptanalysis is also a powerful attack against a block cipher. It reduces the complexity of the exhaustive key search. This attack model works based on the chosen-plaintext. It was first successfully applied on DES [6]. For this attack, we used the same procedure used in linear cryptanalysis, and we discovered an $R-1$ (15) round high probable differential trail of an R (16) rounds cipher [13]. Here, we used the Difference Distribution Table (DDT) of the active S-boxes to find the probabilities of the differential trail of the cipher. A differential trail is said to be optimal if it contains a minimum number of active S-boxes. A DDT of the S-box shows the differential probability for all the possible pairs of the input and output differences. We observed that the DDTs of all the 16 S-boxes of our cipher have the same differential probability for all the possible pairs of the input and output differences. Because of this, the key-dependent non-linear (S-box) layer called the encoding function does not need to differentiate among DDTs of various S-boxes while arriving at the optimal differential trail.

Let two 128-bit plaintexts (inputs) to the system be X' and X'' with the corresponding outputs (outputs of the 15^{th} round of the proposed cipher) as Y' and Y'', respectively. The input difference is denoted by $\triangle X = X' \oplus X''$, and the output difference is denoted by $\triangle Y = Y' \oplus Y''$, where $\oplus$ is a bit-wise addition modulo 2 operation. Both $\triangle X$ and $\triangle Y$ are also 128 bits. The pair $(\triangle X, \triangle Y)$ is referred to as an expected differential characteristic (or an optimal differential trail), if a particular output difference $\triangle Y$ occurs given a particular input difference $\triangle X$ with a suitably high probability. This expected differential characteristic can be arrived at by concatenating the appropriate differential characteristics of the S-boxes from round to round as illustrated in [13]. This is because S-boxes are the only non-linear components of the proposed cipher, and a differential trail consists of a sequence of input and output differences between the rounds. It follows that the output difference from one round corresponds to the input difference for the next round. Generally, arriving at this expected differential characteristic is impractical because we have to process 2^{128} differential characteristics. So, we divide 128-bit input difference block $\triangle X$ into 4 sub-blocks as $\triangle X_1, \triangle X_2, \triangle X_3$, and $\triangle X_4$, where the size of each sub-block is 32 bits. And then, we find all the corresponding differential trails for all the possible non-zero values for the first sub-block $\triangle X_1$ keeping

the remaining three sub-blocks fixed. This gives rise to a maximum of 2^{32} differential trails. Similarly, we repeated the same procedure for the sub-blocks $\triangle X_2$, $\triangle X_3$, and $\triangle X_4$.

We investigate the construction of a differential trail by examining non-zero input differences corresponding to the S-boxes from the 1^{st} round to the 15^{th} round. Using the DDTs of the S-boxes in each round, if a particular non-zero input difference occurs then the corresponding output difference is decided with suitably high probability and least hamming weight. We only consider the S-boxes that have non-zero input differences and hence non-zero output differences to estimate the number of active S-boxes of a differential trail. We used a computer-based search to find an optimal differential trail by evaluating the number of active S-boxes at each round of the proposed cipher, and we found that the minimum number of active S-boxes in an optimal differential trail is 141. Also, it is determined that the proposed cipher's S-boxes have 2^{-2} maximum differential probability. Therefore the differential probability for a 15 rounds cipher is bounded by 2^{-282}. Since the complexity of the R rounds cipher against differential cryptanalysis is inversely proportional to its largest differential probability [13], it follows that the complexity of the proposed cipher against differential cryptanalysis would be 2^{282}. Hence, the proposed cipher with a 128-bit key is resistant to differential cryptanalysis.

6.3 Avalanche effect

One of the desirable properties of a block cipher is that it should exhibit a good avalanche effect, also called the diffusion effect. For a small change in the plaintext difference, there should be a large difference in the ciphertext. Also, a good avalanche effect ensures that the diffusion effect of a block cipher is at least 50%. We looked into the avalanche effects of our proposed block cipher based on plaintexts with low and high hamming weights. The test is as follows: A 128 bits plaintext block B with a low hamming weight that consists of all binary $0's$ (0X00) is considered and we generated 128 plaintexts B_i that differ in 1 bit from the original plaintext B. That is

$$B_i = B \oplus (1 <<_i), \tag{14}$$

where $\oplus$ is the bit-wise `XOR` operation, $<<_i$ is the left shift operation by i bit positions and $0 \leq i \leq 127$.

Now let C be the ciphertext of the original plaintext B and C_i be the ciphertext of the plaintext B_i for $0 \leq i \leq 127$. As part of the test, we calculated the hamming distances between C and C_i in percentages as

$$HDP_i = \frac{D(C, C_i)}{\text{SIZE}(C)} \times 100\%, \tag{15}$$

where $0 \leq i \leq 127$, $D(C, C_i)$ denotes the hamming distance between C and C_i, and SIZE(C) denotes the number of binary digits in the ciphertext C.

We repeated the same process for another 128 bits plaintext with high hamming weight consisting of all binary $1's$ (0XFF), and we calculated all the corresponding values of HDP_i for $0 \leq i \leq 127$. For both the plaintexts, we compared the HDP_i values of our proposed cipher with those of the other existing ciphers such as AES-128 and the ones given in [4, 5, 30]. The corresponding results are shown in Table 8. In the table, the first column shows the values of HDP_i in the specified range; separately; the second, third, fourth, and the fifth columns show the number of times hamming distances (HDP_i) of the ciphertexts $C_0, C_1, \ldots, C_{127}$ from C lie in the specified range given in the first column of the table corresponding to [4, 5], [30], proposed cipher, and AES-128, respectively. Also given in the last two rows of the table are the average (mean) hamming distance in percentage and median absolute deviation (MAD), respectively. The MAD tells us how far the hamming distances from the mean are. From these values, we can conclude that the avalanche effect of our cipher is approximately the same as AES-128, and better than that of all the other existing ciphers given in [4, 5, 30].

Table 8 Results of hamming distances.

Range of HDP_i	Number of ciphertext for [4, 5]		Number of ciphertext for [30]		Number of ciphertext for proposed cipher		Number of ciphertext for AES-128	
	0X00	0XFF	0X00	0XFF	0X00	0XFF	0X00	0XFF
≤ 34.99	0	0	23	32	0	0	0	0
$35 - 44.99$	19	24	38	24	17	16	14	17
$45 - 54.99$	94	84	66	51	91	98	97	103
$55 - 64.99$	15	20	1	21	20	14	17	8
≥ 65	0	0	0	0	0	0	0	0
Mean:	49.37		39.32		49.77		49.52	
MAD:	3.94		13.75		3.34		3.36	

6.4 Strict avalanche criterion (SAC)

A strict avalanche criterion is a generalization of the avalanche effect. It is used to measure the impact on each bit of the ciphertext by changing the bits of the plaintext. If there is a slight change in the plaintext, then the impact of this change on each bit of the corresponding ciphertext should be uniform. That is, whenever a single bit of the plaintext is changed (from 1 to 0 or from 0 to 1), each of the ciphertext's bits changes with a probability of approximately 50% [10]. In order to test whether the proposed cipher meets this criterion, we generated 128 random secret keys and using each of these secret keys we encrypted 1024 different randomly generated plaintexts of the same length. Then, we changed a particular bit in each of these plaintexts (the bit with the same sequence number in all plaintexts), we encrypted these changed plaintexts with each secret key that we generated and then compared them with the original ciphertexts to see how they differ from each other. We repeated this process 128 times (where 128 is the length of each plaintext

used), so that every single bit in each of these plaintexts is changed in the process. The partial results of this experiment are shown in Table 9. Each cell of the table represents the change percentage of the q^{th} bit of the ciphertext when the p^{th} bit of the plaintext is changed, where p is the row number and q is the column number of the table. For example, we can verify in the table that when the 7^{th} bit of the plaintexts is changed, then the 16^{th} bit of the ciphertexts changed in half of the ciphertexts (or 50%).

According to the experimental results, as shown in Table 9, we can conclude that when an arbitrary bit of the plaintexts is changed, then each bit of the ciphertext is changed with the probability of approximately 50%. This means that if a single bit is changed in all of the 1024 plaintexts, then each of the ciphertext's bits will change in approximately half of the ciphertexts. This result implies that the proposed cipher satisfies the strict avalanche criterion. Also, for this test, if we compare the results of our proposed block cipher with the results of BCMPQ, as illustrated in [10], we can observe that the performance of the proposed cipher is better than BCMPQ.

Table 9 Strict avalanche criterion of the proposed cipher.

Plaintext bits (input bits)	Ciphertext bits (output bits)							
	1	2	4	8	16	32	64	128
1	49.97%	50.07%	50.04%	50.14%	49.87%	49.90%	50.08%	50.04%
2	49.83%	49.94%	49.87%	50.04%	49.87%	50.03%	50.17%	49.95%
3	50.09%	49.93%	50.02%	49.82%	49.67%	49.94%	49.97%	49.92%
4	49.97%	49.94%	49.87%	49.78%	49.94%	49.88%	50.00%	49.76%
5	50.02%	50.05%	50.07%	49.80%	49.87%	49.76%	50.04%	49.73%
6	49.65%	50.00%	50.04%	50.01%	50.08%	49.89%	49.93%	50.00%
7	50.35%	50.02%	50.21%	50.30%	50.00%	49.95%	50.06%	49.93%
8	50.21%	49.89%	49.99%	50.09%	49.71%	50.26%	50.12%	49.95%
16	49.83%	50.07%	50.04%	50.30%	50.06%	49.91%	49.86%	49.79%
20	49.88%	49.80%	50.34%	49.95%	50.14%	49.94%	49.87%	50.00%
25	50.17%	50.04%	49.90%	50.13%	49.87%	50.00%	50.07%	49.58%
32	50.24%	50.08%	50.01%	49.94%	49.76%	49.95%	49.78%	49.74%
40	50.09%	50.17%	49.97%	49.90%	50.07%	50.07%	50.05%	50.10%
41	50.00%	50.17%	49.89%	50.07%	49.98%	49.91%	50.06%	49.89%
64	50.11%	50.15%	50.06%	49.99%	50.13%	49.90%	50.01%	49.93%
100	50.05%	50.14%	50.02%	49.90%	50.20%	50.26%	49.82%	49.91%
110	50.20%	49.87%	49.86%	50.00%	50.08%	50.10%	49.98%	49.90%
120	50.03%	50.00%	50.02%	49.85%	49.94%	49.96%	50.13%	49.90%
127	50.03%	49.79%	50.06%	50.00%	50.08%	49.84%	50.13%	49.89%
128	49.87%	49.80%	50.16%	50.08%	49.92%	50.00%	50.04%	50.17%

6.5 Statistical test for randomness

The ciphertexts created using our proposed cipher pass various statistical tests of NIST-STS. Using the NIST-STS [2] suite we evaluated the randomness of the obtained ciphertexts. Each test of the NIST-STS package gives a P-value and Success/Fail status. The P-value is the probability that a perfect random

[2]National Institute of Technology - Statistical Test Suite

number generator would have produced a less random sequence than the one being tested [23]. Also, in NIST-STS, each test is given a P-value threshold, also called the significance level α. If P-value $\geq \alpha$, the sequence is considered random; otherwise, non-random. Typically, the value of α is chosen in the range [0.001, 0.01]. We have used NIST Spec. Publ. 800-22 rev. 1a package with $\alpha = 0.01$ that consists of 15 types of statistical tests [23]. For each of these tests, we randomly chose 128 bits IV, 128 bits secret key, and 8192 128 bits plaintext blocks. A binary sequence of 1048576 bits is constructed using the ciphertext obtained in the CBC mode. That is, it is a binary sequence obtained by concatenating the 8192 ciphertext blocks of 128 bits each. We generated 1000 such binary sequences for the same plaintext blocks using different random 128 bits keys. We ran each of these tests on the outputs of both the proposed cipher and AES-128 1000 times and compared the randomness of the proposed cipher with that of the AES-128 for each binary sequence of 1048576 bits. Table 10 shows these NIST-STS experimental results. Column A of the table lists the names of the tests carried out. The number of accepted binary sequences, the number of rejected binary sequences, and the proportion of binary sequences that passed a statistical test at the $\alpha = 0.01$ significance level for both the proposed cipher and AES-128 are listed in columns B and C, respectively. As a result, the randomness of the proposed cipher is comparable to that of the AES-128. Hence, from the NIST-STS's point of view, both the ciphers are random.

Table 10 For 1000 random keys, results of the NIST test for the proposed encryption as compared to AES-128 encryption system when the same key is used for both cryptosystems with CBC mode of operation.

A	B			C		
Tests	Proposed			AES-128		
	Number of success	Number of failures	Proportion of success out of 1000	Number of success	Number of failures	Proportion of success out of 1000
Frequency	993	7	0.993	991	9	0.991
Block frequency	983	17	0.983	991	9	0.991
Cumulative sum	994	6	0.994	992	8	0.992
Runs	990	10	0.990	989	11	0.989
Longest run	990	10	0.990	992	8	0.992
Rank	996	4	0.996	991	9	0.991
DFT	982	18	0.982	986	14	0.986
Non-overlapping template	985	15	0.985	980	20	0.980
Overlapping template	992	8	0.992	994	6	0.994
Universal statistical	985	15	0.985	988	12	0.988
Approximate entropy	991	9	0.991	993	7	0.993
Random excursion	989	11	0.989	985	15	0.985
Random excursion variants	991	9	0.991	990	10	0.990
Serial	993	7	0.993	994	6	0.994
Linear complexity	995	5	0.995	981	19	0.981

7 Conclusion and future work

This paper has proposed an efficient block cipher to encrypt or decrypt data in the form of a block of 128 bits. The proposed cipher uses 16 optimal S-boxes of 4×4 bits that form an optimal quasigroup. The design of the proposed cipher is based on the concept of multiple quasigroups. We have analyzed our cipher for several attacks, including linear cryptanalysis and differential cryptanalysis. We found that our cipher is resistant to these attacks. Also, we have analyzed the software performance (time complexity), space complexity, and avalanche effect (diffusion effect) of our cipher by comparing it with AES-128 and other existing quasigroup based block ciphers [4, 5, 30]. We noted that the avalanche effect of our cipher is almost the same as that of AES-128 and due to more computations our cipher is slightly slower than AES-128, but our cipher uses half the space compared to AES-128. Also, the proposed block cipher uses the same amount of space as that used by [30] but 512 times lesser than [4, 5]. We also noted that our cipher is more efficient than DES. In addition, our cipher is more than 2 times faster and gives a better avalanche effect than other existing quasigroup based block ciphers [4, 5, 30]. Hence, we concluded that our cipher appears to be an excellent alternative for the quasigroup based proposals. We have also analyzed our cipher against the strict avalanche criterion (SAC). The results show that when a random bit of plaintext is changed, the proposed cipher changes each bit of the ciphertext with a probability of approximately 50%. Hence the proposed cipher satisfies the SAC.

Remember that our cipher works as a family of block ciphers parameterized by an optimal quasigroup. So, if required, the security of the cipher can be enhanced by keeping the optimal quasigroup secret along with the secret key. That is, the security of the proposed cipher depends not only on the secret key but also on the optimal quasigroup employed. Note that our cipher uses 16 optimal quasigroups in all the 16 rounds, and these 16 quasigroups are generated from the initial optimal quasigroup by circularly shifting the rows. Since an optimal quasigroup is constructed using the 16 optimal S-boxes of 4×4 bits, we, therefore, can form a maximum of 16! optimal quasigroups by permuting the rows. So, the total key space of our cipher would then be $C_1^{16!} \times 16^{16} \times 2^{128} \approx 2^{236}$. Therefore, our cipher can be seen to be more secure than AES-128 against quantum attack since in quantum computing [12], the best quantum attack against any symmetric-key cryptosystem is proportional to the square root of the key space. And, the attack complexity of our cipher against quantum attack is about 2^{118}, while in the case of AES-128 is only 2^{64}.

The randomness of the obtained ciphertexts produced by the proposed cipher is tested using the NIST statistical test suite. We ran our encryption system for a random plaintext of 1048576 bits with 1000 different keys and generated 1000 ciphertexts. The results of the proposed cipher are compared with that of AES-128 for the same plaintext and the same keys. We observed that the randomness of the outputs of our cipher and AES-128 are comparable to each other.

In future, we intend to cryptanalyze our cipher against impossible differential cryptanalysis, zero-correlation linear cryptanalysis, and related-key attack.

References

[1] Ashwini, P., Venkaiah, V. C., Bhukya, W. N. : Secret sharing scheme based on Latin squares. Journal of Discrete Mathematical Sciences & Cryptography, Taylor and Francis: 1–15. doi: 10.1080/09720529.2021.1925447 (2021)

[2] Banik, S., Pandey, S. K., Peyrin, T., Sasaki, Y., Sim, S. M., Todo, Y. : GIFT: a small Present-towards reaching the limit of lightweight encryption. Cryptographic Hardware and Embedded Systems–CHES, Lecture Notes in Computer Science, vol 10529. Springer, Cham. doi:10.1007/978-3-319-66787-4_16 (2017)

[3] Basit, A., Chanakya, P., Venkaiah, V.C. and Moiz, S.A.: New multi-secret sharing scheme based on superincreasing sequence for level-ordered access structure, International Journal of Communication Networks and Distributed Systems, Vol. 24, No. 4, pp.357–380 (2020)

[4] Battey, M., Parakh, A. : Efficient quasi-group block cipher for sensor networks. In 21^{st} international conference on computer communications and networks, IEEE, 1–5 (2012)

[5] Battey, M., Parakh, A. : An efficient quasigroup block cipher. Wireless personal communications, 73(1), 63–76 (2013)

[6] Biham, E., Shamir, A. : Differential cryptanalysis of DES-like cryptosystems. Journal of CRYPTOLOGY, 4(1), 3-72 (1991)

[7] Chauhan, D., Gupta, I., Verma, R. : Quasigroups and their applications in cryptography. Cryptologia, 45(3), 227–265, doi:10.1080/01611194.2020.1721615 (2000)

[8] Coppersmith, D., Rogaway, P. W. : International Business Machines Corp. Software-efficient pseudorandom function and the use thereof for encryption, U.S. Patent 5,454,039 (1995)

[9] Dénes, J., Keedwell, A. D. : Latin squares: New developments in the theory and applications, Vol. 46, Elsevier (1991)

[10] Dimitrova, V., Kostadinoski, M., Trajcheska, Z., Petkovska, M., Buhov, D.: Some Cryptanalysis of the Block Cipher BCMPQ. ICT Innovations 2014 Web Proceedings, 201–209 (2014)

[11] Fugaku, S.: A64FX 48C 2.2GHz, Tofu interconnect D, Fujitsu RIKEN Center for Computational Science Japan (2016)

[12] Grover, L. K.: A fast quantum mechanical algorithm for database search. In Proceedings of the twenty-eighth annual ACM symposium on Theory of computing, pp. 212-219 (1996)

[13] Heys, H. M.: A tutorial on linear and differential cryptanalysis. Cryptologia, 26(3), 189–221 (2002)

[14] Josuttis, N. M.: The C++ standard library: a tutorial and reference, 2^{nd} Edition (2002)

[15] Koscielny, C.: Generating quasigroups for cryptographic applications. International journal of applied mathematics and computer science, 12(4), 559–570 (2002)

[16] Kumar, U., Venkaiah, V. C.: A New Modified MD5-224 Bits Hash Function and an Efficient Message Authentication Code Based on Quasigroups. In Cyber Security, Privacy and Networking Proceedings of ICSPN 2021, LNNS, vol 370, Springer, 1–12 (2022)

[17] Kumar, U., Agarwal, A., Venkaiah, V. C.: New Symmetric Key Cipher Based on Quasigroup. In Cyber Security, Privacy and Networking Proceedings of ICSPN 2021, LNNS, vol 370, Singapore, Springer, 83–94 (2022)

[18] Leander, G., Poschmann, A.: On the classification of 4 bit S-Boxes. In proceedings of the 1^{st} international workshop on arithmetic of finite fields, Springer-Verlag, 159–176 (2007)

[19] Van, L., Hendricus, J., Wilson, R. M., Wilson, R. M.: A course in combinatorics, Cambridge university press (2001)

[20] Matsui, M.: Linear cryptanalysis method for DES cipher. Advances in cryptology, Proc. Eurocrypt'93, LNCS 765, T. Helleseth, Ed., Springer-Verlag, 386–397 (1994)

[21] Meyer, K. A.: A new message authentication code based on the non-associativity of quasigroups. Iowa State University (2006)

[22] Mihajloska, H., Gligoroski, D.: Construction of Optimal 4-bit S-boxes by Quasigroups of Order 4. In the sixth international conference on emerging security information, systems and 163–168 (2012)

[23] Rukhin, A., Soto, J., Nechvatal, J., Smid, M., Barker, E., Leigh, S., Levenson, M., Vangel, M., Banks, D., Heckert, A., Dray, J., Vo, S.: A statistical test suite for random and pseudorandom number generators for cryptographic applications NIST. Special, Publication, 800-22, Revision 1a (2001)

[24] Schneier, B.: Description of a new variable-length key, 64-bit block cipher (Blowfish). In International Workshop on Fast Software Encryption, Springer, Berlin, Heidelberg, 191-204 (1993)

[25] Schneier, B.: Applied cryptography-protocols, algorithms, and source code in C. John Wiley & sons (2007)

[26] Selvi, D., Velammal, G., Arockiadoss, T.: Modified method of generating randomized latin squares. IOSR journal of computer engineering (IOSR-JCE), 16(1), 76–80 (2014)

[27] Shcherbacov ,V.: Elements of quasigroup theory and some its applications in code theory and cryptology. Lectures in Prague, Czech Republic (2003)

[28] Shcherbacov, V.: Quasigroup based crypto-algorithms, arXiv preprint arXiv:1201.3016 (2018)

[29] Stones, R. J., Su, M., Liu, X., Wang, G., Lin, S.: A Latin square autotopism secret sharing scheme. Designs, codes and cryptography, 80(3), 635–650 (2016)

[30] Zhao, Y. , Xu, Y.: A Lightweight block cipher based on quasigroups. In 6th International Conference on Advanced Materials and Computer Science (2017a)

[31] Zhao, Y. , Xu, Y.: Optimal S-boxes based on 3-quasigroups of order 4. In 6th International Conference on Advanced Materials and Computer Science (2017b)