Journal of Information Security and Applications 76 (2023) 103514

Contents lists available at ScienceDirect

Journal of Information Security and Applications

journal homepage: www.elsevier.com/locate/jisa

Conception of efficient key-dependent binary diffusion matrix structures for dynamic cryptographic algorithms

Hassan N. Noura [a,*], Ola Salman [b], Ali Chehab [b]

[a] Univ. Bourgogne Franche-Comté (UBFC), FEMTO-ST Institute, France
[b] American University of Beirut, Electrical and Computer Engineering Department, Beirut 1107 2020, Lebanon

ARTICLE INFO

Keywords:
Binary diffusion matrix
Dynamic diffusion primitives
Dynamic cryptographic algorithms
Flexible diffusion matrix
Fixed points
Linear branch number

ABSTRACT

The existing static cryptographic algorithms suffer from different performance and security challenges. These limitations are attributed to their fixed structure, where the substitution and diffusion primitives maintain the same values throughout the process. In this paper, we present a new framework of a dynamic binary diffusion matrix with flexible dimensions ($n \times n$). The proposed solution replaces the static diffusion primitives, in existing symmetric and un-keyed cryptographic algorithms, with dynamic primitives. We define six different dynamic and flexible binary diffusion forms, four that are invertible, and two that are non-invertible, and hence, they could be used for different security services. However, using a single diffusion form does not always guarantee the required cryptographic properties. To that effect, we propose a binary multiplication scheme of a dynamic primary matrix and its transposed form, which yields the desired efficiency, and provides good resistance against recent implementation attacks, yet without degrading the system performance. We conduct security and performance analyses to validate the effectiveness of the proposed solution. The results confirm the cryptographic performance in terms of the linear branch number and the number of fixed points. In this context, the best-obtained branch number is 4 for $n = 8$ and 5 for $n = 12$ for the invertible forms, while for the non-invertible ones, the best branch number is 3 for $n = 8$ and 4 for $n = 12$. Moreover, in terms of the number of fixed points, the obtained numbers are very close to 0 for the invertible and non-invertible forms.

1. Introduction

The security of any un-keyed or Symmetric Cryptographic Algorithm (SCA) highly depends on the performance of its diffusion and substitution primitives. The diffusion operation is essential for the required diffusion property and to achieve the avalanche effect. The design of a robust diffusion process has attracted the attention of many security researchers since block ciphers and coding schemes rely on an invertible diffusion layer, while a non-invertible one can be used for stream ciphers, hash functions, key derivation functions, and Pseudo Random-Number Generators (PRNG) [1]. Typically, the structure of a round function either follows that of the Substitution–Permutation Networks (SPN) or the Feistel Networks (FN) [2].

The majority of existing SCA standards adopt the static structure since the proper selection of the substitution and diffusion layers ensures the maximum cryptographic performance against traditional attacks. Even though static substitution and diffusion layers have better immunity against traditional attacks, however, it is not the case when it comes to emerging attacks such as physical attacks [3]. On the other hand, a dynamic SCA adopts a key-dependent structure where the substitution and diffusion layers are variable, and depend on a secret key. An example of a dynamic SCA is the Blowfish technique [4], which relates the construction of the substitution layers to the secret key. This paper focuses on the design of the diffusion operation, which is a linear transformation with a matrix representation, either over the binary Galois Field $GF(2)$ or using integers over $GF(2^m)$, where m represents the precision (number of bits) of the matrix elements.

1.1. Related work

Different types of diffusion layers have been presented in the literature (see Table 3) such as the Maximum Distance Separable (MDS) codes that were originally introduced in error-correcting codes. They are called maximum distance separable because of the property where any two codewords in the code have the maximum possible Hamming distance, which is the number of bits in which they differ. This property motivated cryptographers to use it to ensure strong diffusion, as in the Advanced Encryption Standard (AES) [5], which uses it in the MixColumns operation. The MDS matrix used in AES is a (4×4) matrix

* Corresponding author.
E-mail address: hassan.noura@univ-fcomte.fr (H.N. Noura).

https://doi.org/10.1016/j.jisa.2023.103514

Available online 6 June 2023

over the finite field $GF(2^8)$. MDS is applied on a column of the input data and mixes its bits together in a way that helps to spread them out across the entire column, which provides diffusion at the column level, and in the avalanche effect in combination with the shift row permutation operation [5] after several rounds.

There is also the **Binary Matrix (BM)** approach, which has an advantage over MDS since its implementation requires only XOR operations while MDS (consumes more than 75% of AES execution time) requires, in addition to XOR operations, *xtime* function calls (or table look-ups) [6].

Modern algorithms attempt to satisfy the diffusion property by relying on a binary form such as Camellia [7] with a matrix size of (8×8) and ARIA [8] with a size of (16×16). Both types of matrices exhibit the maximum linear branch number (5 and 8, respectively), and thus, they are known as Maximum Distance Binary Linear (MDBL) codes [9]. To the best of our knowledge, all of the existing BM candidates **have a static nature** [10]-[14], except for the ones presented in our previous work [15]. In [10], the authors presented a static method to build a (16×16) binary matrix with a branch number equals to 8. Then, they extended the previous work in [11] by defining a new construction method for (32×32) BMs with a branch number equals to 10. Recently, a similar approach was presented in [12] to build (8×8) and (16×16) static matrices with maximum branch numbers. Additionally, other static BMs for lightweight block ciphers and hash functions were investigated in [13,14]. However, invertible BMs cannot be used in the design of hash functions due to the required one-way property of a hash function. Hence, there is a need for the design of non-invertible BMs.

Cryptographic algorithms can be separated into two groups depending on the underlying cryptographic primitives, static or dynamic:

1. Static primitives: They represent the majority of algorithms. They have fixed substitution and diffusion primitives that are selected based on their optimal cryptographic performance. The selected S-box should ensure a minimal differential probability (against differential attacks), and a minimal linear probability (against linear attacks). In addition, these algorithms employ static diffusion that features a high linear branch number and a low number of fixed points. In terms of performance, static cryptographic algorithms require multiple rounds to achieve the desired cryptographic properties, which introduces overhead in terms of delay and resources. This can be a hard challenge for real-time applications and/or limited devices. Even though such algorithms guard against analytic attacks, however, they are vulnerable to a set of implementation attacks such as side-channel attacks, in addition to an expensive computational complexity that might are not suitable for limited devices or real-time applications. Note that there is high computational overhead when introducing a countermeasure against a side-channel attack, which may not be practical.
2. **Dynamic primitives**: Following significant advancements in cryptanalysis, the need for a new kind of cryptographic algorithm has become imperative. These solutions must be lightweight and guarantee a high level of security against both current and advanced attacks (such as current cryptanalysis and implementation techniques or future ones). In this work, we argue and we prove that one possible solution is to rely on dynamic cryptographic primitives [16], which are not fixed primitives, but that are updated frequently, either per round, per block, or per message. Note that dynamic primitives do not typically offer the maximum required cryptographic properties in terms of linear/differential probability and branch number, against cryptanalysis as static ones, however, their importance and usefulness stem from the fact that they are unknown and variable compared to static public ones. This introduces a hard challenge for attackers, who have to know which set of diffusion and substitution primitives are used for the collected ciphertext. It is important to emphasize that the dynamic cryptographic approach does not lend itself to formal security proofs. Thus, a heuristic study is required to show that the majority of the produced cryptographic primitives (substitution and diffusion) satisfy the desired cryptographic properties.

1.2. Problems and challenges

Existing symmetric cryptographic algorithms can resist the different cryptanalysis techniques such as linear, differential, or brute force attacks that try to exploit weaknesses at the algorithm level. However, the hardware implementations of these ciphers, with their current structure, are vulnerable to side-channel attacks. These attacks exploit the system's physical properties or characteristics (such as power consumption, electromagnetic radiation, or the emitted sound) that are leaked from a device during its operation, rather than exploiting a flaw in the system's software or in the cryptographic algorithm. Designers of symmetric ciphers need to incorporate appropriate countermeasures at the algorithm level in addition to software and hardware implementations to prevent these side-channel attacks.

The vulnerability of static symmetric ciphers against side-channel attacks is mainly due to the static cryptographic primitives that are iterated for several rounds. Hence, they exhibit fixed physical properties such as energy consumption or electromagnetic radiation profile, which could be exploited by attackers performing for example, power analysis or differential power analysis. On the other hand, variable cryptographic primitives exhibit random physical properties of power, electromagnetic radiation, and emitted sound. As such, dynamic ciphers complicate the process of side-channel attacks. Also, in the event of a side-channel attack, a countermeasure need only be introduced at the algorithm level. This is the main driver behind our work to propose binary diffusion structures that would help in designing new dynamic cipher algorithms.

The existing diffusion structures have been designed to be static, invertible, and with a fixed size. To the best of our knowledge, the work in [15] was the first to propose the design of a dynamic, flexible, binary, and invertible diffusion layer, however, the proposed approach exhibits limitations in terms of performance. In this paper, a new dynamic approach is proposed for higher performance, where efficiency is proven and compared to the well-known diffusion layers in the literature. Currently, there is no existing approach that exhibits good performance in a pseudo-random manner, and modern cryptographic algorithms require a minimum efficiency when relying on a dynamic structure. Thus, the main contribution of this paper stems from the design of a new dynamic binary diffusion scheme that achieves good cryptographic performance with acceptable overhead. The cryptographic performance is quantified based on the traditional diffusion cryptographic metrics, which are the linear branch number and the number of fixed points.

1.3. Contribution

This work targets the design of a construction technique for a new type of invertible and non-invertible binary diffusion matrices that are dynamic and flexible. The invertible diffusion matrices could be used in block ciphers, and channel coding algorithms such as network coding. The non-invertible dynamic primitive, with the one-way property, could be used in stream ciphers, hash functions, and pseudo-random number generators.

In this paper, we propose a set of algebraic integer and binary diffusion matrix forms to construct dynamic diffusion matrices. Moreover, these matrices are theoretically derived and adapted to the binary Galois field from the integer Galois field.

The advantage of the presented work concentrates on defining a new kind of diffusion matrices with the following properties:

Table 1
Table of notation.

Symbol	Definition
n	Dimension of the diffusion matrix
m	bits precision that can be 8, 16, 32, and 64.
$x \odot y$	The proposed adapted binary matrix multiplication between two matrices x and y
$B \circ X$	The binary diffusion process on input X using the diffusion matrix B that can be invertible or not-invertible
$\oplus$	The bit-wise addition, modulo 2, of two-bit strings of equal length
BM	Invertible binary diffusion matrix
NBM	Non-invertible binary diffusion matrix
M_u	The first pseudo-sub matrix with size of $\frac{n}{2} \times \frac{n}{2}$.
M_v	The second pseudo-random sub-matrix with size of $\frac{n}{2} \times \frac{n}{2}$
$I_m = I_l$	Identity matrix with size of $\frac{n}{2} \times \frac{n}{2}$

Table 2
List of abbreviation.

Abbreviation	Explanation
SCA	Symmetric Cryptographic Algorithm
PRNG	Pseudo Random-Number Generators
SPN	Substitution–Permutation Network
FN	Feistel Networks
AES	Advanced Encryption Standard
BM	Binary Matrix
NBM	Non-invertible Binary Matrix
MDS	Maximum Distance Separable
MDBL	Maximum Distance Binary Linear
NI	Non-invertible diffusion Matrix
BN	Branch Number
No	Number of Operations

- A dynamic structure based on a dynamic key.
- Invertible and non-invertible matrices (according to the required cryptographic task).
- A binary form to simplify the hardware and software implementations.
- Flexibility in terms of dimensions and precision.
- Good cryptographic properties with high branch number and low fixed points.

Each of the proposed primary forms requires two sub-matrices to be constructed (Mu and Mv). However, the first three primary matrix forms, on their own, do not ensure the best possible cryptographic performance. Hence, we propose a scheme with a binary matrix multiplication between the primary matrix forms and their transposed ones to produce the final $(n \times n)$ diffusion matrix, either invertible or non-invertible. The majority of the resulting binary matrices exhibit high cryptographic performance when compared to a single primary form. From a performance perspective, the computational complexity for the generation of the final diffusion matrix is relatively low; it is simply based on the iteration of a stream cipher to construct two matrix forms, in addition to the binary matrix multiplication in the Galois binary field. This does not require any optimization, and it provides a high-security level. Furthermore, the security and performance results validate the effectiveness and robustness of the proposed binary diffusion construction approach for a set of forms.

1.4. Organization

The rest of this paper is organized as follows. Section 2 reviews a relevant dynamic diffusion layers and discusses the challenges for ensuring good cryptographic performance. Section 3 presents the proposed dynamic technique to build invertible binary diffusion matrices with high cryptographic performance. In Section 4, the different non-invertible binary diffusion forms are proposed and discussed, while in Section 5, the performance of the proposed solution is analyzed. Finally, Section 6 concludes this work and discusses its perspectives.

2. Preliminary

In this section, we discuss the necessary background information on the dynamic key-dependent cryptographic algorithms, and the dynamic diffusion primitives (matrix forms). Moreover, the security tests, which are used to quantify the robustness of diffusion primitives, are presented. This section enhances the readability and understanding of the proposed approach. Note that all notations used in this paper are shown in Table 1. Also, a list of abbreviations is presented in Table 2.

2.1. Diffusion primitives

The binary diffusion layer has been proven to be a good candidate due to its low computational complexity and simple implementation, which only requires the logical "Exclusive OR" operation (XOR). The challenge is to provide a new diffusion layer that is binary and with high cryptographic performance. In fact, the Number of Possible Vectors (NPV) that form an $(n \times n)$ invertible matrix over field q was presented in [17] and presented in Eq. (1):

$$\prod_{k=1}^{n}(q^n - q^{k-1}) \tag{1}$$

In addition, the variation of the number of possible vectors in function of n is illustrated in Fig. 1-a

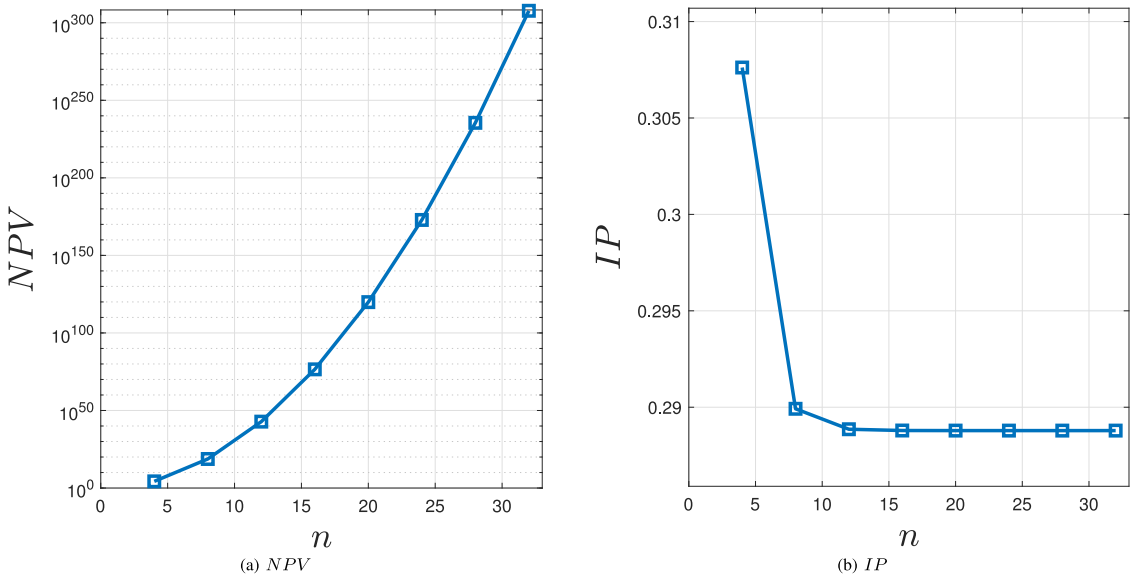
So the Invertibility Probability (IP) of this matrix can be computed by using Eq. (2):

$$\frac{\prod_{k=1}^{n}(q^n - q^{k-1})}{q^{n^2}} = \prod_{k=1}^{n}(1 - q^{k-1-n}) < 1 - \frac{1}{q} \tag{2}$$

However, this probability decreases as n grow, limiting the flexibility property. Also, when $q = 2$, the invertibility probability of a binary matrix (2×2) converges to ≈ 0.288788 as $n \to \infty$ [17], as shown in Fig. 1-b. Therefore, there are much fewer invertible matrices in the binary field in comparison to the integer field. This means that a good bound cannot be obtained in the binary Galois field. Also, a large q (e.g. 2^8, 2^{16}) guarantees that the elements of the diffusion vector are linearly independent. Their high probability is close to 0.9961 for $q = 2^8$, but they require a large number of arithmetic operations. Accordingly, they would consume more than one clock cycle for each operation, which reduces the efficiency of the diffusion layer.

The proposed approach aims to address these limitations, and it is based on the work presented in [18], where four dynamic integer invertible forms of diffusion layers were presented, each requiring two sub-matrices in the construction process. However, these forms have a limitation since the existing sub-matrices do not have equal priorities; there is always one sub-matrix that has higher values compared to the other sub-matrices. This limitation was then solved in [16] by applying a shuffling algorithm to the lines (rows) and then, to the columns for each diffusion matrix. However, this solution was adapted for secret encoding packets; it uses integer matrix multiplication and a shuffling algorithm, and hence a relatively high computational complexity. To address the latter challenge, dynamic integer, and binary forms were presented in [15], against which we analyze and compare our proposed technique. The results show that the proposed matrix forms outperform those presented in [15].

The dynamic binary diffusion form of [15] suffers from poor cryptographic performance. Thus, we define a new diffusion matrix form by mapping integer diffusion matrix forms, such as that in [18,19], into the binary field.



(a) NPV (b) IP

Fig. 1. Variation of the possible number of vectors (NPV) that form an $n \times n$ binary matrix (a) and the invertibility probability (IP) (b) versus n.

2.2. Dynamic key-dependent cryptographic algorithms

Traditional symmetric cryptographic algorithms, such as AES, have a static structure: the substitution table has constant values, and the multiplication of columns is carried out always with the same diffusion matrix. Throughout the encryption/decryption processes, these two primitives maintain the same values.

The dynamic cryptographic approach consists of the use of a secret key with a nonce to produce a dynamic key, which is then used to produce dynamic cryptographic primitives. For each new message, or set of messages (depending on configuration), the dynamic key changes, and so do the cryptographic primitives used in the encryption process. Thus, the substitution and diffusion primitives vary and they are no longer known to the attackers, which enhances the robustness level against attacks.

For this reason, there is a current trend to rely on the dynamic approach in the design of cryptographic algorithms [20–23]. This work is based on the concept of dynamic cryptographic primitives to satisfy the confusion and diffusion properties in cryptographic algorithms. In addition, we make use of the dynamic mode of operation, presented in [24], where blocks are selected in a pseudo-random manner for each input message.

2.3. Dynamic binary diffusion matrix structures

The proposed diffusion matrix forms have binary values instead of integer ones, which simplifies the arithmetic operations (addition and multiplication) that are performed using just logical operations (XOR). The binary diffusion process, as shown in Algorithm 1, uses a $(n \times n)$ binary diffusion matrix B, an input data vector of n elements, $X = \{x_1, x_2, \ldots, x_n\}$, and produces an output diffused vector $DX = \{dx_1, dx_2, \ldots, dx_n\}$ with n elements. Note that $b_{i,j}$ represents the diffusion element at the ith row and the jth column.

For example, for $n = 4$, the binary diffusion operation is expresses by the following equation:

$$DX_{4\times1} = B_{4\times4} \circ X_{4\times1}, \tag{3}$$

$$\begin{bmatrix} dx_1 \\ dx_2 \\ dx_3 \\ dx_4 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{bmatrix} \circ \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} x_1 \oplus x_2 \oplus x_4 \\ x_2 \oplus x_4 \\ x_2 \\ x_3 \oplus x_4 \end{bmatrix}$$

Algorithm 1 Binary diffusion process ($B\circ$ X)

```
procedure BINARY_MIXING(X, B)
    n ← length(X)
    DX ← zeros(n)
    for i ← 1 to n do
        for j ← 1 to n do
            if b_{i,j} ≠ 0 then
                dx_i ← dx_i ⊕ x_j
            end if
        end for
    end for
    return DX
end procedure
```

where $\circ$ represents the binary diffusion process described in Algorithm 1, $\oplus$ is the XOR operation, B is an example of an invertible (4×4) binary diffusion matrix, X is a (4×1) input vector which includes 4 elements, x_1, x_2, x_3 and x_4, and DX is the resulting diffused vector which has 4 elements.

Accordingly, in the binary diffusion process, the computational complexity of the diffusion process is decreased tremendously. The principal contribution is in the design of efficient and flexible key-dependent binary diffusion matrix forms (invertible and non-invertible) while maintaining the security robustness of the cryptographic primitives.

In the following, we explain the security metrics used to quantify the robustness of the produced binary diffusion matrices, the linear branch number, and the fixed points [2].

2.4. Security tests

Two metrics have been used to measure the efficiency of the proposed invertible and non-invertible diffusion matrix forms: (1) the branch number [25] and (2) the number of the fixed points [26].

2.4.1. Branch number

The branch number of a diffusion matrix represents the diffusion rate and quantifies its security against traditional attacks such as linear and differential cryptanalysis. Additionally, it denotes the minimum number of active elements (different) for any two consecutive rounds.

The diffusion process is a linear transformation that can be represented by a matrix. Let n be the number of input/output elements for the diffusion process D, where the size of the input and output of each element is m bits. Furthermore, the diffusion process can be defined as $D : (\{0,1\}^m)^n \rightarrow (\{0,1\}^m)^n$. The branch number BN of an $(n \times n)$ diffusion matrix D is calculated according to Eq. (4):

$$BN(D) = min\{wt(x) + wt(D \cdot x^t)\} | x \neq 0 \tag{4}$$

where $()^t$ represents the transposed matrix, $x = \{x_1, x_2, \ldots, x_n\} \in (\{0,1\}^m)^n$, $x_i \in \{0,1\}^m$, $i = 1, 2, \ldots, n$ and $wt(c)$ denotes the Hamming weight of a code word c (the number of non-zero bits in c).

2.4.2. Number of fixed points

The effect of the number of Fixed Points (FP) of the diffusion operation is given in [26] and it can be computed using Eq. (5):

$$FP(D) = \sum_{x=0}^{2^m} [x = D \cdot x^t] \tag{5}$$

where $x = \{x_1, x_2, \ldots, x_n\} \in (\{0,1\}^m)^n$ and $FP(D)$ represents the number of non-changing elements, after the diffusion process, using the diffusion matrix D. In fact, when the number of fixed points of a linear transformation greatly exceeds the expected number of a random linear transformation, this results in poor diffusion since the bits in these blocks are left unchanged when producing the output blocks. Also, note that the expected number of fixed points in a random permutation is one [26].

3. Proposed invertible diffusion forms

The construction process of the binary diffusion matrix starts by converting the integer diffusion matrices into binary. Also, we change the minimum number of required diffusion forms for good cryptographic performance. The same properties of the integer forms (flexibility, invertibility, dynamicity) are inherited by the binary forms.

In the following, we propose four invertible binary diffusion forms, where each is based on a specific rule. Note that these primary matrix forms require pseudo-random square sub-matrices, such as Mu and Mv. We present, for each proposed matrix form, the cryptographic results in terms of branch number and fixed points, for $n = 8$ and 12. In general, the results show that mixing a primary diffusion matrix with a transposed one achieves better performance compared to using a single primary diffusion form. However, these results are variable, and they depend on the generated sub-matrices (Mu, Mv). Also, the results vary between acceptable and high values.

3.1. First invertible diffusion matrix form

This form was investigated in [18,19], where integer diffusion forms, have determinant values equal to 1 (non-singular matrix). In the following, the four invertible diffusion integer matrix forms M_1, M_2, M_3, and M_4 with a size of $(n \times n)$ of [18,19] are presented.

$$M_1 = \begin{bmatrix} I_m & Mu \\ Mv & I_l + Mu \cdot Mv \end{bmatrix}, \quad M_2 = \begin{bmatrix} Mu & I_m \\ I_l + Mu \cdot Mv & Mv \end{bmatrix},$$
$$M_3 = \begin{bmatrix} I_l + Mu \cdot Mv & Mv \\ Mu & I_m \end{bmatrix}, \quad M_4 = \begin{bmatrix} Mv & I_l + Mu \cdot Mv \\ I_m & Mu \end{bmatrix}$$

Two sub-matrices Mu and Mv are required to form each of the above diffusion matrices. The "$\odot$" represents an integer matrix multiplication as described in [27,28].

The adaptation of the integer forms, M_w, $w = 1, 2, 3, 4$, to provide a binary diffusion layer (BM_w) can be obtained by applying the following modifications:

(1) Employing Binary sub-matrices (Mu and Mv) instead of integer ones.

Algorithm 2 The proposed binary matrix multiplication algorithm ($\odot$)

```
procedure Z = Modified_Matrix_Binary_MultiPlication(X, Y)
    [m, n] = size(X)
    q = NumberofColumn(Y)
    for i ← 1 to m do
        for j ← 1 to q do
            tmp ← 0
            for w ← 1 to n do
                tmp ← tmp ⊕ (X(i, k) ∧ Y(k, j))
            end for
            Z(i, j) ← tmp
        end for
    end for
    return Z
end procedure
```

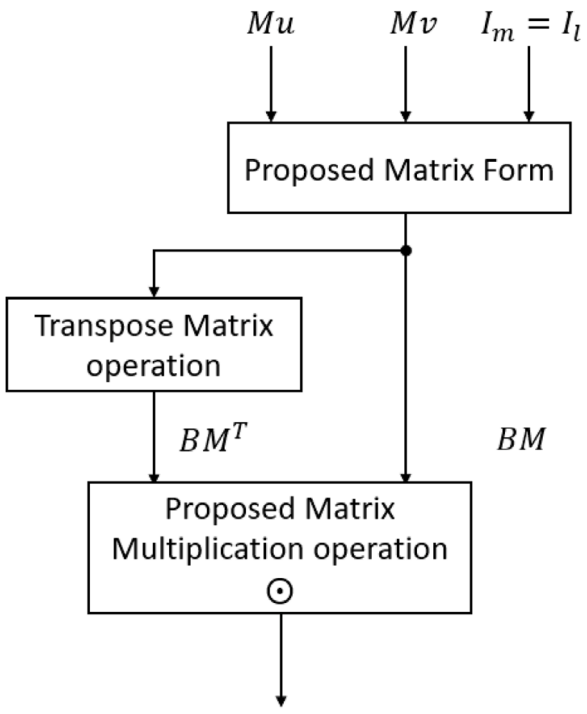


Fig. 2. The proposed approach to construct binary diffusion matrix $D = BM \odot BM^T$ using the first invertible matrix forms by multiplying the primary matrix with its transposed one. (I_m and I_l are required for a set of primary diffusion matrix forms).

(2) Replacing the addition and subtraction arithmetic operations by the logical Exclusive-OR ($\oplus$) operation;
(3) Performing matrix multiplication in the binary Galois field using logical (XOR, AND) operations ($\oplus$; $\wedge$);

The proposed modified matrix multiplication ($\odot$) in the binary field is described in Algorithm 2.

Therefore, the obtained binary diffusion forms with their corresponding inverse versions are given in Eq. (6)–(9):

$$BM_1 = \begin{bmatrix} I_m & Mu \\ Mv & I_l \oplus (Mv \odot Mu) \end{bmatrix},$$
$$BM_1^{-1} = \begin{bmatrix} I_l \oplus (Mu \odot Mv) & Mu \\ Mv & I_m \end{bmatrix} \tag{6}$$

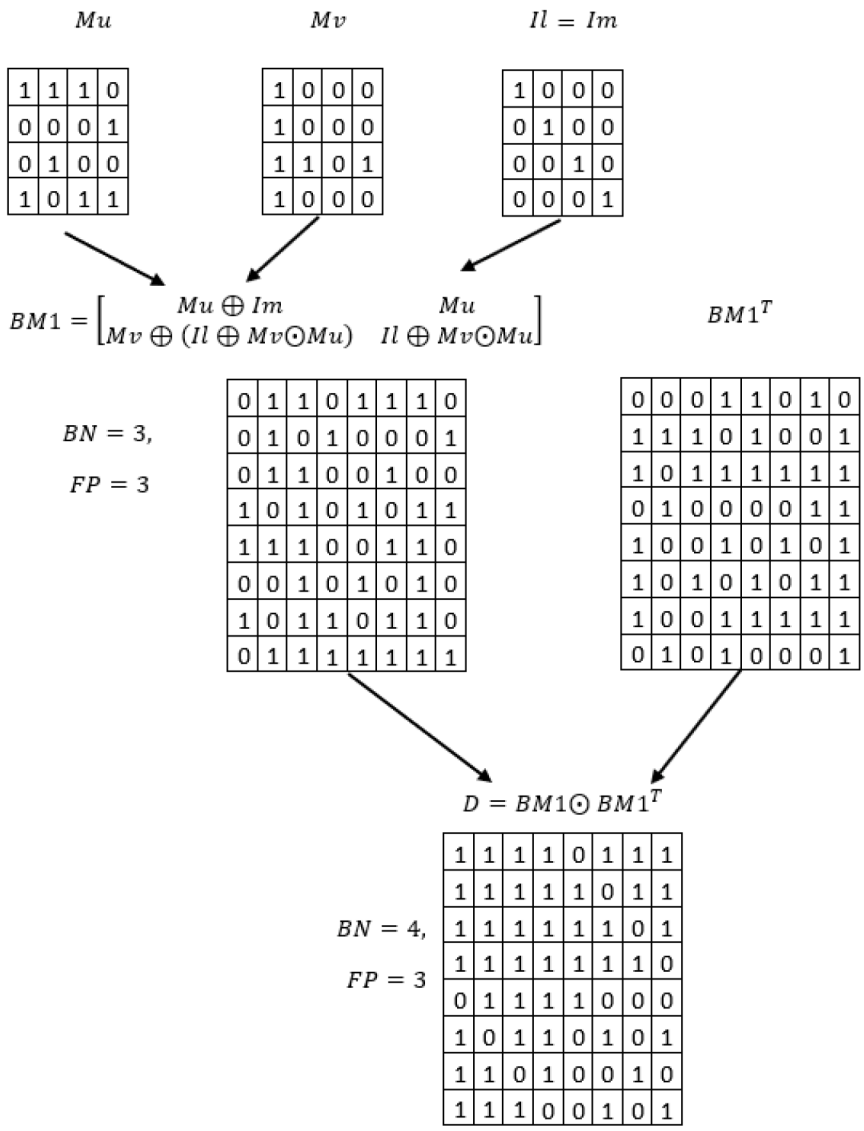


Fig. 3. A numerical example to construct a binary diffusion matrix using the first invertible matrix forms.

$$BM_2 = \begin{bmatrix} Mu & I_m \\ I_l \oplus (Mv \odot Mu) & Mv \end{bmatrix}, \quad BM_2^{-1} = \begin{bmatrix} Mv & I_m \\ I_l \oplus (Mu \odot Mv) & Mu \end{bmatrix} \tag{7}$$

$$BM_3 = \begin{bmatrix} I_l \oplus (Mv \odot Mu) & Mv \\ Mu & I_m \end{bmatrix}, \quad BM_3^{-1} = \begin{bmatrix} I_m & Mv \\ Mu & I_l \oplus (Mu \odot Mv) \end{bmatrix} \tag{8}$$

$$BM_4 = \begin{bmatrix} Mv & I_l \oplus (Mv \odot Mu) \\ I_m & Mu \end{bmatrix}, \quad BM_4^{-1} = \begin{bmatrix} Mu & I_l \oplus (Mu \odot Mv) \\ I_m & Mv \end{bmatrix} \tag{9}$$

Note that BM_2, BM_3 and BM_4 are shifted versions of BM_1.

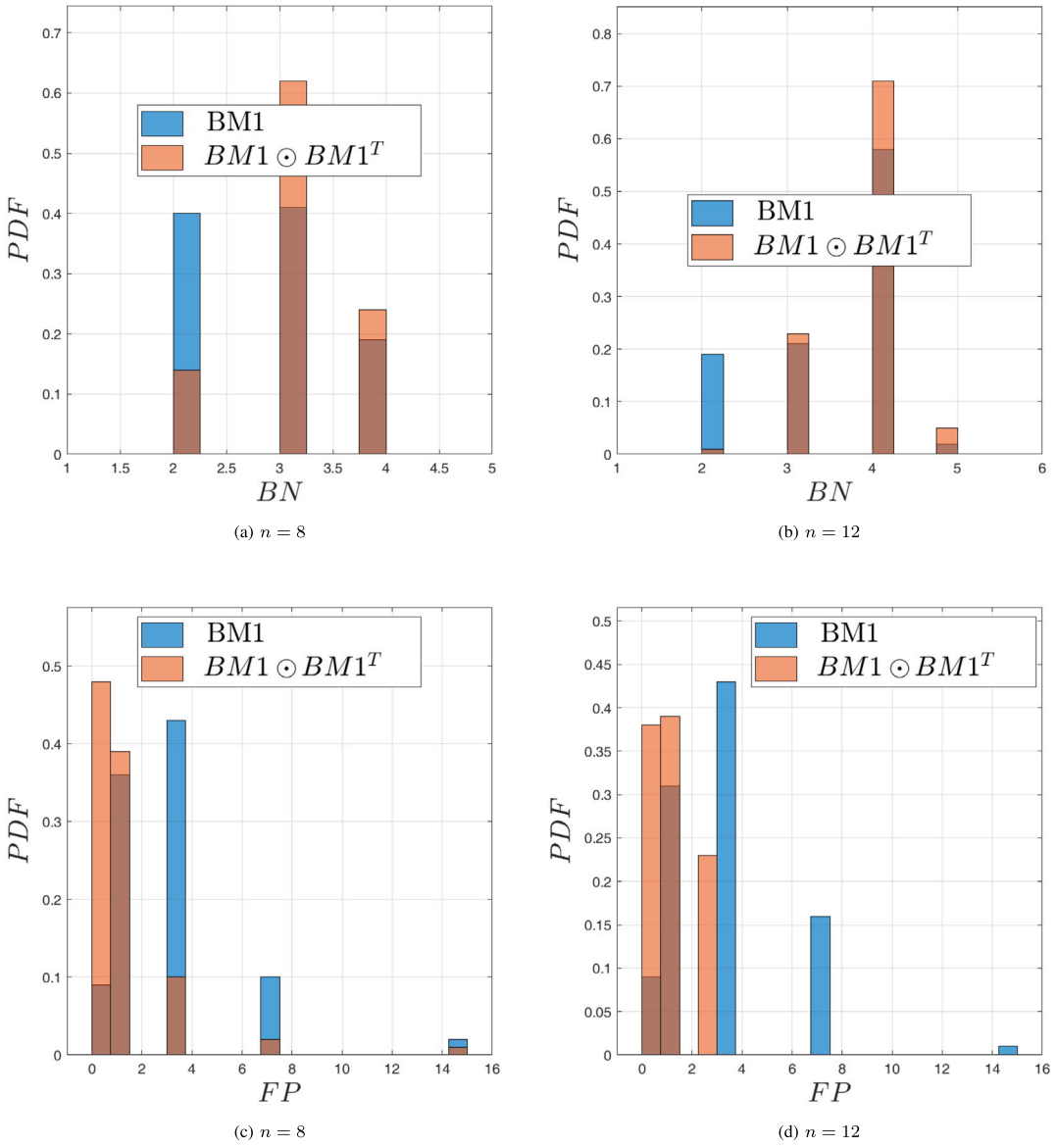
To avoid having identity sub-matrices, we propose combining the two columns of each matrix BM using the XOR operation to enhance the cryptographic performance of the corresponding diffusion layers. The resulting forms become:

$$BM_1 = \begin{bmatrix} Mu \oplus I_m & Mu \\ Mv \oplus I_l \oplus (Mv \odot Mu) & I_l \oplus (Mv \odot Mu) \end{bmatrix}, \quad BM_1^{-1} = \begin{bmatrix} I_l \oplus (Mu \odot Mv) & Mu \\ Mv \oplus I_l \oplus (Mu \odot Mv) & Mu \oplus I_m \end{bmatrix} \tag{10}$$

$$BM_2 = \begin{bmatrix} Mu & Mu \oplus I_m \\ I_l \oplus (Mv \odot Mu) & Mv \oplus I_l \oplus (Mv \odot Mu) \end{bmatrix}, \quad BM_2^{-1} = \begin{bmatrix} Mv \oplus I_l \oplus (Mu \odot Mv) & Mu \oplus I_m \\ I_l \oplus (Mu \odot Mv) & Mu \end{bmatrix} \tag{11}$$

$$BM_3 = \begin{bmatrix} I_l \oplus (Mv \odot Mu) & Mv \oplus I_l \oplus (Mv \odot Mu) \\ Mu & Mu \oplus I_m \end{bmatrix}, \quad BM_3^{-1} = \begin{bmatrix} Mu \oplus I_m & Mv \oplus I_l \oplus (Mu \odot Mv) \\ Mu & I_l \oplus (Mu \odot Mv) \end{bmatrix} \tag{12}$$

$$BM_4 = \begin{bmatrix} Mv \oplus I_l \oplus (Mv \odot Mu) & I_l \oplus (Mv \odot Mu) \\ Mu \oplus I_m & Mu \end{bmatrix}, \quad BM_4^{-1} = \begin{bmatrix} Mu & I_l \oplus (Mu \odot Mv) \\ Mu \oplus I_m & Mv \oplus I_l \oplus (Mu \odot Mv) \end{bmatrix} \tag{13}$$



(a) $n = 8$ (b) $n = 12$

(c) $n = 8$ (d) $n = 12$

Fig. 4. Probability Density Function (PDF) of the BN (a)–(b) and FP (c)–(d) by using the proposed first matrix forms with $n = 8$ and 12, respectively.

where BM_i and BM_i^{-1}, for $i = 1, 2, 3, 4$, always have determinants equal to one. The binary matrix multiplication ($\odot$) over the Galois binary field is described in Algorithm 2. Fig. 2 illustrates the steps to construct a primary diffusion matrix and its enhanced version, while Fig. 3 shows a numerical example of this construction process for the primary matrix (BM_1) and the enhanced one, D. The cryptographic performance, in terms of fixed points and linear branch number, for each of these primary diffusion forms does not always offer the required results of the cryptographic metrics, as it will be shown in Fig. 4 for $n = 8$ and 12. As such, we propose to apply the binary matrix multiplication between any primary diffusion matrix and its transposed one. The results in Fig. 4 show that this enhances the density of branch numbers and fixed points; the density of low branch numbers is decreased and that of high branch numbers is increased. Also, the density of high fixed points is decreased while the density of low fixed points is decreased. In addition, a numerical representation, of three binary invertible matrices using the first invertible forms, is shown in Fig. 5, for $n = 8$, 12, 16 that have branch number equals to 4, 5, and 6, respectively.

Moreover, increasing n leads to enhanced cryptographic robustness (e.g., the higher value of branch numbers is increased, and the maximum value of fixed points is reduced) as shown in Fig. 4-(c) and (d) compared to Fig. 4-(a) and (b), respectively.

3.2. Second invertible diffusion matrix form

This form is based on a specific rule; we use, in each primary matrix, an upper or lower triangular binary matrix with the diagonal part equal to one, to ensure invertibility. These upper or lower matrix elements can be generated in a pseudo-random manner. Fig. 6 and 7 show numerical examples of the primary and the enhanced one by using the second matrix forms; they use upper and lower pseudo-random triangular invertible binary matrices, respectively.

Fig. 8 shows the density of branch numbers and fixed points, for $n = 8$ and 12, for the primary matrices and the enhanced ones by using the proposed second matrix forms. The results show clearly the improvement of the enhanced matrices over the original ones. Originally, all primary matrices have a very low branch number equal to 2,

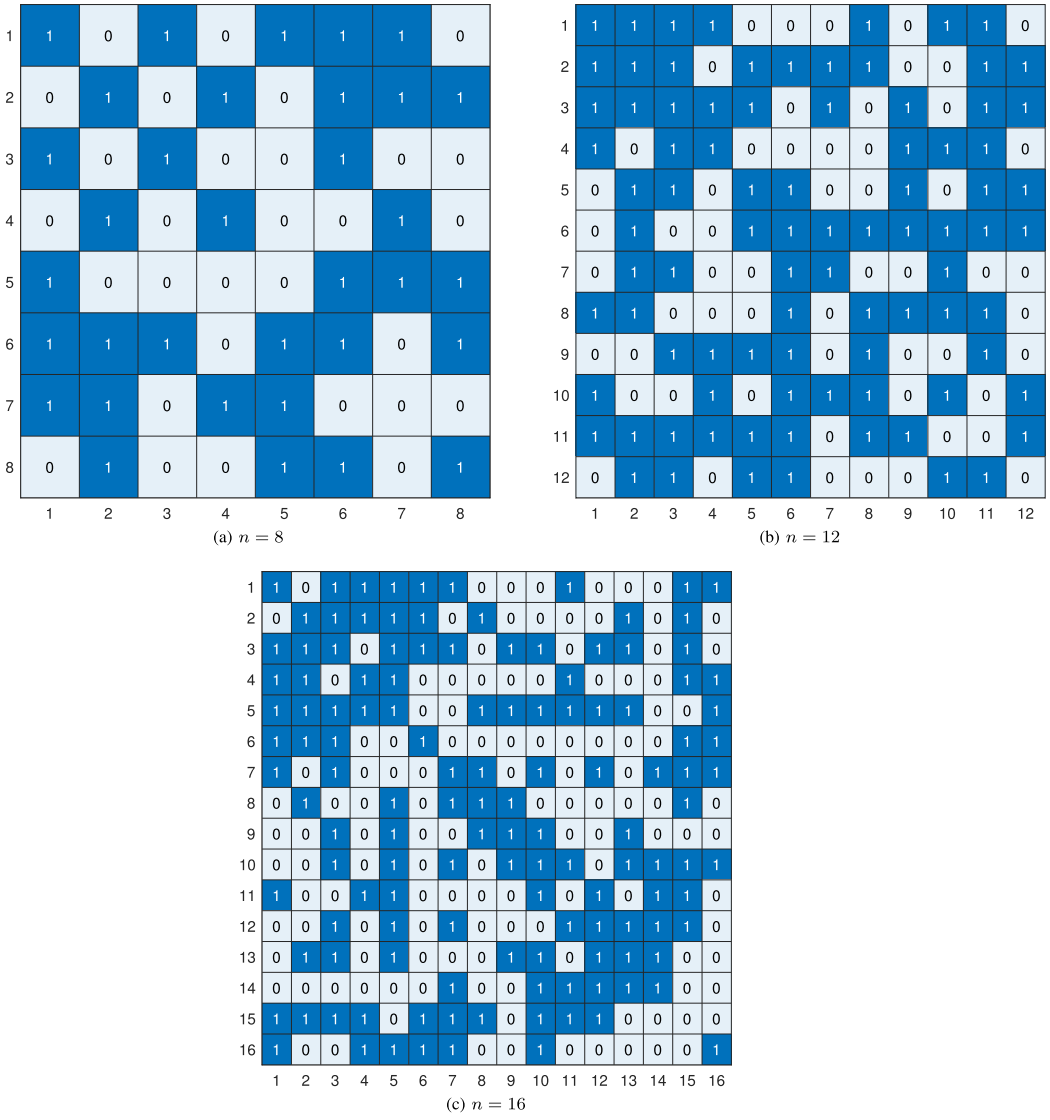


Fig. 5. A numerical representation of three binary invertible matrices, for (a) $n = 8$, (b) $n = 12$, and (c) $n = 16$. The matrices have branch number equals to 4 for (a), 5 for (b), and 6 for (c) by using the proposed enhanced scheme with the first invertible forms.

independent of n. However, the enhanced scheme increases the branch number range and reduces the density of the smallest branch number (2). Furthermore, the majority of the produced diffusion matrices, via the enhanced scheme, have a greater branch number: the values are 3 and 4, for $n = 8$; and 3, 4, and 5, for $n = 12$. On the other hand, these matrices show a reduction in the number of fixed points and their probabilities.

3.3. Third invertible diffusion matrix form

In this sub-section, we propose the third invertible diffusion matrix form (four diffusion matrices). First, we consider a general matrix G which is constructed from four sub-matrices A, B, C and D, as shown below:

$$G = \begin{bmatrix} A & B \\ C & D \end{bmatrix} \tag{14}$$

For matrix G to be invertible it should have a determinant equal to one, which is the determinant of identity matrix I. Hence,

$$AD - BC = I,$$
$$AD = I + BC, \tag{15}$$

For this form, we can consider that $I = C \odot C^{-1}$ or $I = B \odot B^{-1}$. In case of $I = C \odot C^{-1}$, this leads to:

$$AD = CC^{-1} + BC \tag{16}$$
$$= C(C^{-1} + B) \tag{17}$$

Consequently, we get $A = C$, $A^{-1} = C^{-1}$, and $D = C^{-1} + B$.

The final obtained form will be:

$$G = \begin{bmatrix} C & B \\ C & C^{-1} + B \end{bmatrix} \tag{18}$$

The determinant of the corresponding matrix is calculated as follows,

$$\begin{aligned} det(G) &= det(A) \times det(D - C \times A^{-1} \times B) \\ &= det(C) \times det(C^{-1} + B - C \times C^{-1} \times B) \\ &= det(C \times C^{-1}) \\ &= det(I) \\ &= 1 \end{aligned} \tag{19}$$

where $det(A \cdot B) = det(A) \cdot det(B)$.

Two squared sub-matrices (Mu and Mv) are required to construct each diffusion matrix form of the third proposition. These sub-matrices

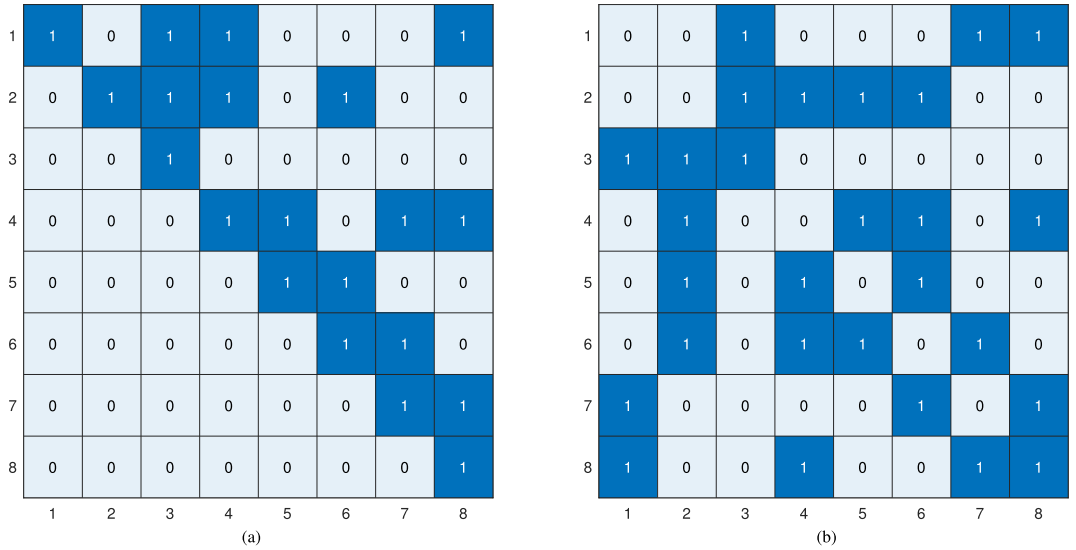


Fig. 6. Numerical examples for the second matrix forms: An upper pseudo-random triangular invertible binary matrix U (a) for $n = 8$ ($BN = 2$, and 7 fixed elements) and the corresponding final diffusion matrix $D = U \odot U^T$ (b) that has $BN = 4$ and without any fixed elements.

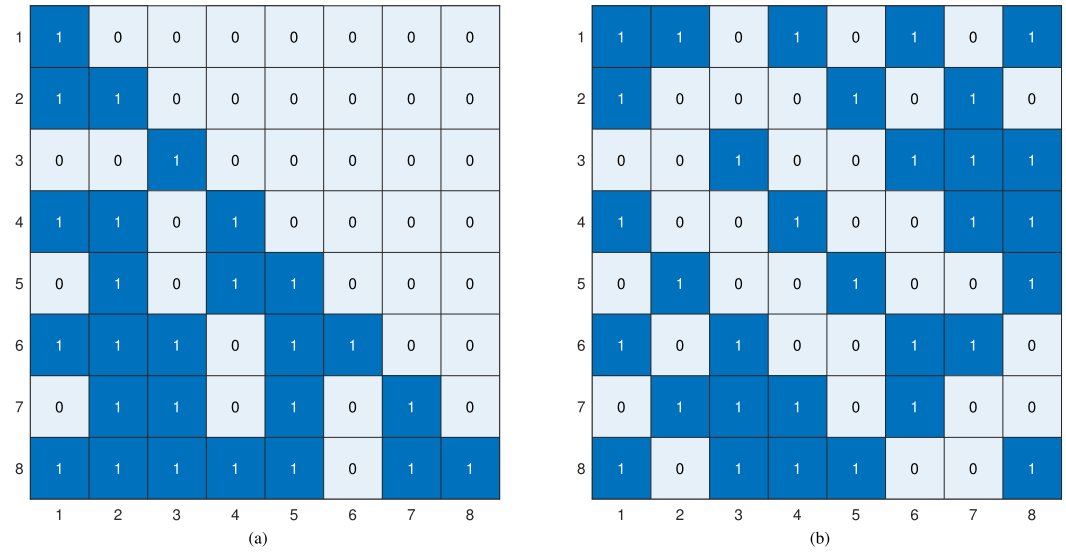


Fig. 7. Numerical examples for the second matrix forms: A lower pseudo-random triangular invertible binary matrix L (a) for $n = 8$ ($BN = 2$, and 7 fixed elements) and the corresponding final diffusion matrix $D = L \odot L^T$ (b) that has $BN = 4$ and without any fixed elements.

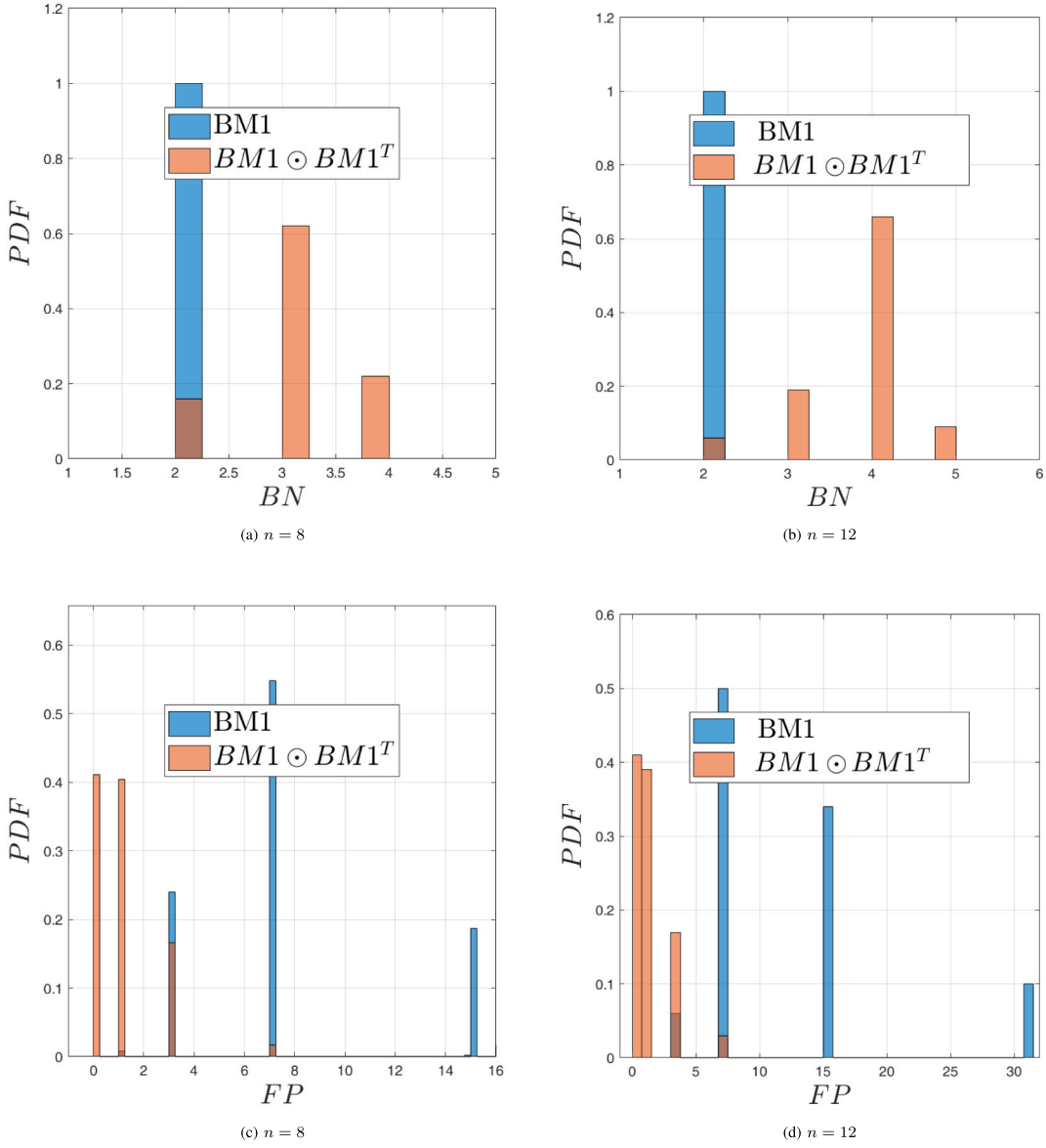
can be produced in a pseudo-random manner by using a deterministic pseudo-random generator. In a similar manner to the previous matrix forms, four flexible, dynamic, integer, primary matrix forms can be constructed with a size of $(n \times n)$. Here, $(\odot)$ represents the matrix multiplication. The difference between the third matrix forms and the previous ones is that the sub-matrix Mv should have an inverse sub-matrix Mv^{-1}. Therefore, the previous first or second matrix forms can be used to construct the invertible sub-matrix Mv. Also, an identity matrix is not required for the third matrix forms. Replacing sub-matrices C and B in the general matrix form with Mv and Mu, yields to:

$$BM_1 = \begin{bmatrix} Mv & Mu \\ Mv & Mv^{-1} \oplus Mu \end{bmatrix},$$
$$BM_2 = \begin{bmatrix} Mv & Mv^{-1} \oplus Mu \\ Mv & Mu \end{bmatrix} \quad (20)$$
$$BM_3 = \begin{bmatrix} Mu & Mv \\ Mv^{-1} \oplus Mu & Mv \end{bmatrix},$$
$$BM_4 = \begin{bmatrix} Mv^{-1} \oplus Mu & Mv \\ Mu & Mv \end{bmatrix} \quad (21)$$

Note that BM_2, BM_3 and BM_4 are shifted versions of BM_1. In order to enhance the security level even further, we update the diffusion matrix form by replacing the first column with the XOR of the original two columns, in each matrix, to get the following final forms:

$$BM_1 = \begin{bmatrix} Mv \oplus Mu & Mu \\ Mv \oplus Mv^{-1} \oplus Mu & Mv^{-1} \oplus Mu \end{bmatrix},$$
$$BM_2 = \begin{bmatrix} Mv \oplus Mv^{-1} \oplus Mu & Mv^{-1} \oplus Mu \\ Mv \oplus Mu & Mu \end{bmatrix} \quad (22)$$
$$BM_3 = \begin{bmatrix} Mu & Mv \oplus Mu \\ Mv^{-1} \oplus Mu & Mv \oplus Mv^{-1} \oplus Mu \end{bmatrix},$$
$$BM_4 = \begin{bmatrix} Mv^{-1} \oplus Mu & Mv \oplus Mv^{-1} \oplus Mu \\ Mu & Mv \oplus Mu \end{bmatrix} \quad (23)$$

Fig. 9 shows the density of branch numbers and fixed points of these third primary matrix forms. The cryptographic results are those of the final binary diffusion matrix (the multiplication of each primary matrix with its transposed one). We can see that the density of branch number 4 has increased, but the density of fixed points has increased from the



(a) $n = 8$ (b) $n = 12$

(c) $n = 8$ (d) $n = 12$

Fig. 8. Distribution of the BN (a)–(b) and FP (c)–(d) by using the proposed second matrix forms with $n = 8$ and 12, respectively.

value 0 for the original matrices. We conclude that for this variant, the original forms outperform the proposed enhanced versions.

3.4. Fourth invertible diffusion matrix form

Starting with the four diffusion matrices, derived from the third version, and considering that $I = BB^{-1}CC^{-1}$ for Eq. (15), we get:

$$\begin{aligned} AD &= B \cdot B^{-1} \cdot C \cdot C^{-1} + B \cdot C \\ &= B \cdot C(B^{-1} \cdot C^{-1} + I) \end{aligned} \tag{24}$$

As a result, $A = BC$ and $D = B^{-1}C^{-1} + I$. The final obtained form of the third general matrix is,

$$G = \begin{bmatrix} BC & B \\ C & B^{-1}C^{-1} + I \end{bmatrix} \tag{25}$$

The determinant of the produced matrix form is calculated as follows,

$$\begin{aligned} det(G) &= det(A) \& det(D - C \cdot A^{-1} \cdot B) \\ &= det(B \cdot C) \cdot det(B^{-1}C^{-1} + I - C \cdot (BC)^{-1} \cdot B) \\ &= det(B \cdot C) \cdot det(B^{-1}C^{-1} + I - C \cdot C^{-1} \cdot B^{-1} \cdot B) \\ &= det(B \cdot C) \cdot det(B^{-1}C^{-1} + I - I) \\ &= det(B \cdot C) \cdot det(B^{-1}C^{-1}) \\ &= det(B \cdot C) \cdot det(B^{-1}C^{-1}) \\ &= det(I) \\ &= 1 \end{aligned} \tag{26}$$

Again, we replace sub-matrices C and B in the general matrix with Mv and Mu. BM_2, BM_3 and BM_4 are shifted versions of BM_1.

$$\begin{aligned} BM_1 &= \begin{bmatrix} Mu \odot Mv & Mu \\ Mv & (Mu^{-1} \odot Mv^{-1}) \oplus I_l \end{bmatrix}, \\ BM_2 &= \begin{bmatrix} Mv & (Mu^{-1} \odot Mv^{-1}) \oplus I_l \\ Mu \odot Mv & Mu \end{bmatrix} \\ BM_3 &= \begin{bmatrix} Mu & Mu \odot Mv \\ (Mu^{-1} \odot Mv^{-1}) \oplus I_l & Mv \end{bmatrix}, \\ BM_4 &= \begin{bmatrix} (Mu^{-1} \odot Mv^{-1}) \oplus I_l & Mu \\ Mv & Mu \odot Mv \end{bmatrix} \end{aligned} \tag{27, 28}$$

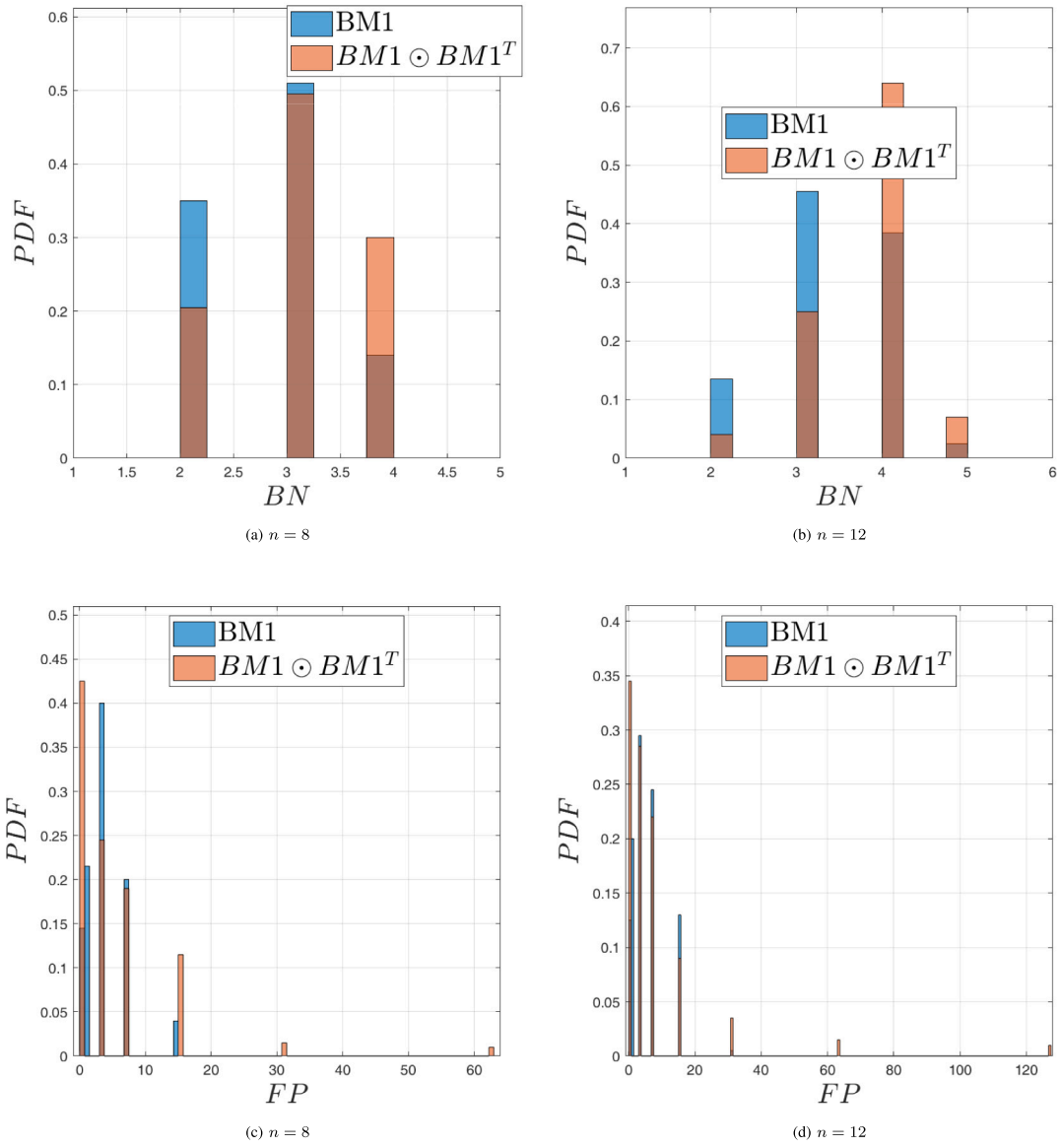


Fig. 9. Distribution of the BN (a)–(b) and FP (c)–(d) for the proposed third matrix forms and the enhanced scheme with $n = 8$ and 12, respectively.

The difference in these matrices, compared to the first and second forms, is that they require both sub-matrices (Mv, Mu) to have inverse sub-matrices (Mv^{-1}, Mu^{-1}), respectively. Therefore, the previous first or second matrix forms can be used to construct the invertible sub-matrices Mv and Mu.

Fig. 10 shows the density of branch numbers and fixed points, for the fourth primary matrix forms, and their corresponding enhanced versions via multiplication with the transposed one. The latter ones show a low density for the higher branch number and fixed points compared to the primary matrix forms (the density of branch number 4 is decreased and the density of fixed points of 0 is also decreased). However, the density of the lower branch number is high with the primary matrix forms compared to enhanced ones. Furthermore, the density of the average branch number is increased with the enhanced ones. We conclude that for this variant, the proposed enhanced technique does not yield better cryptographic performance, and hence, we use the original forms, and there is no need to multiply them with their transpose.

4. Proposed non-invertible diffusion forms

In this section, we present three general forms for Non-Invertible (NI) diffusion matrices. The advantage of this variant is that it ensures the one-way property, which is a principal condition for several cryptographic algorithms such as hash functions and stream ciphers.

4.1. First form of NI diffusion matrix

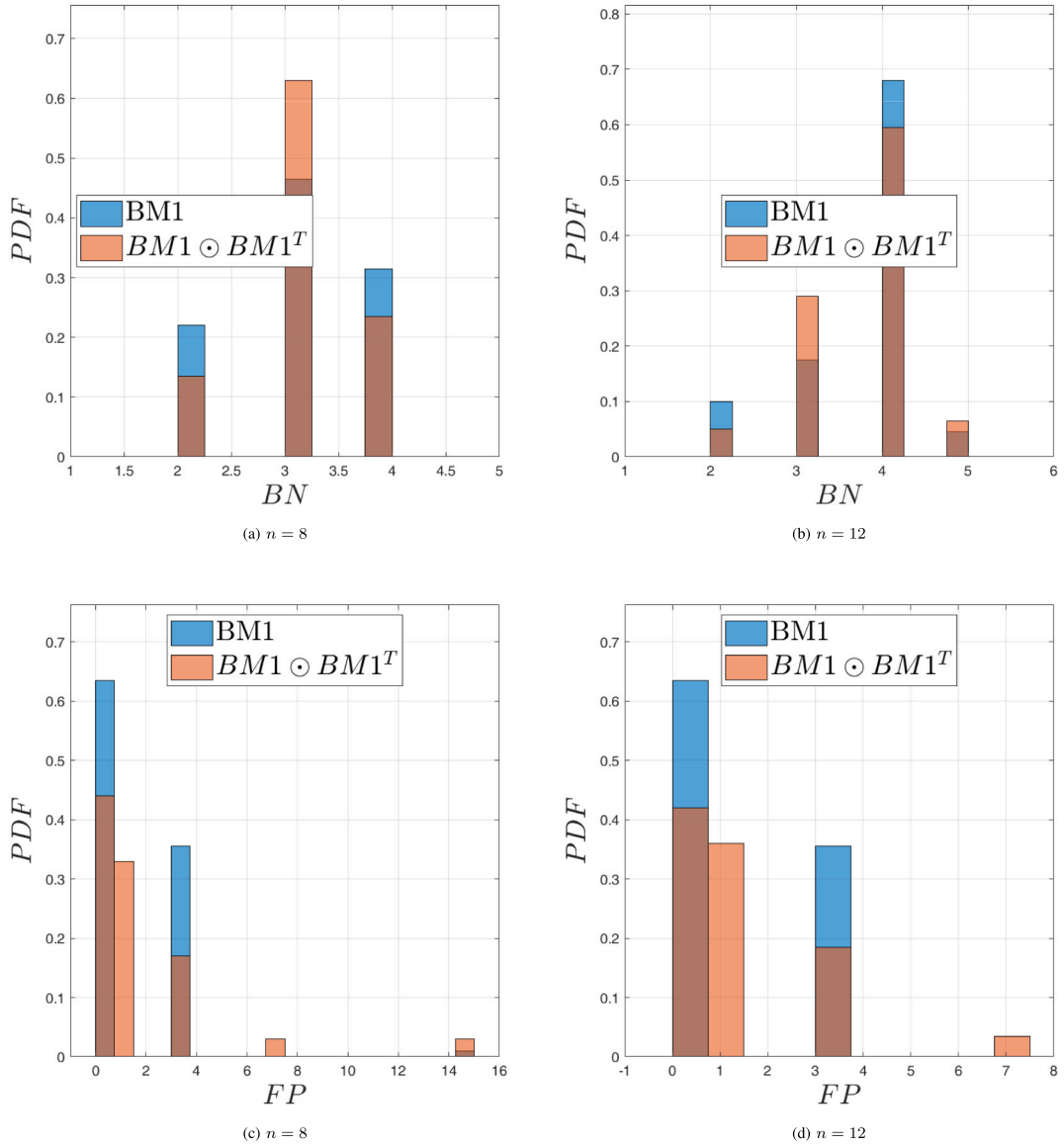
The first non-invertible primary matrix form consists of four flexible diffusion matrices, having a dimension of ($n \times n$), and they are expressed as follows:

$$NBM_1 = \begin{bmatrix} Mu \oplus I_m & Mu \\ Mv \oplus (Mv \odot Mu) & Mv \odot Mu \end{bmatrix},$$

$$NBM_2 = \begin{bmatrix} Mu & Mu \oplus I_m \\ Mv \odot Mu & Mv \oplus (Mv \odot Mu) \end{bmatrix} \quad (29)$$

$$NBM_3 = \begin{bmatrix} Mv \odot Mu & Mv \oplus (Mv \odot Mu) \\ Mu & Mu \oplus I_m \end{bmatrix},$$

$$NBM_4 = \begin{bmatrix} Mv \oplus (Mv \odot Mu) & Mv \odot Mu \\ Mu \oplus I_m & Mu \end{bmatrix} \quad (30)$$



(a) $n = 8$ (b) $n = 12$

(c) $n = 8$ (d) $n = 12$

Fig. 10. Distribution of the BN (a)–(b) and FP (c)–(d) for the proposed fourth matrix forms and the enhanced scheme with $n = 8$ and 12, respectively.

NBM_1, NBM_2, NBM_3, and NBM_4 are primary flexible, non-invertible binary diffusion matrices. NBM_2, NBM_3, and NBM_4 are shifted versions of NBM_1.

Fig. 11 presents the density of BN and FP for the first non-invertible diffusion matrix forms. The maximum branch number obtained with this form is 3 for $n = 8$. A minor enhancement in results is achieved when we multiply this primary matrix with its transpose one.

4.2. Second form of NI diffusion matrix

This form is based on a specific rule, in which we can utilize a non-invertible matrix to ensure the non-invertibility of the produced primary matrix $BM1$. For example, we can use an upper (U) or lower (L) triangular matrix with the diagonal part equal to zero, in order to ensure a determinant equals to 0. Then, this matrix (U or V) will be multiplied by a pseudo-random binary matrix (invertible or non-invertible) RBM.

$$NBM1 = U \odot RBM \text{ or } NBM1 = L \odot RBM \tag{31}$$

Since the upper or lower triangular matrix has a determinant equals to 0, the produced binary matrix has a determinant of 0, and consequently, the non-invertibility property is ensured.

Another possibility to construct a non-invertible matrix, instead of U and L, is by using the following form that requires two pseudo-random sub-matrices Mu and Mv:

$$\begin{bmatrix} Mu & Mv \\ Mu & Mv \end{bmatrix} \tag{32}$$

The results in Fig. 12 show that the primary matrix forms produce non-invertible diffusion matrices with lower branch number than those of the ones multiplied with pseudo-random binary matrix. In addition, fixed points are also low for both cases.

5. Resources and computation analysis

In this section, we analyze the performance of the forms to quantify their effectiveness, mainly in terms of flexibility and memory consumption.

5.1. Computational complexity

The main objective of this work is to produce diffusion matrices that can achieve a high level of security with minimum computational

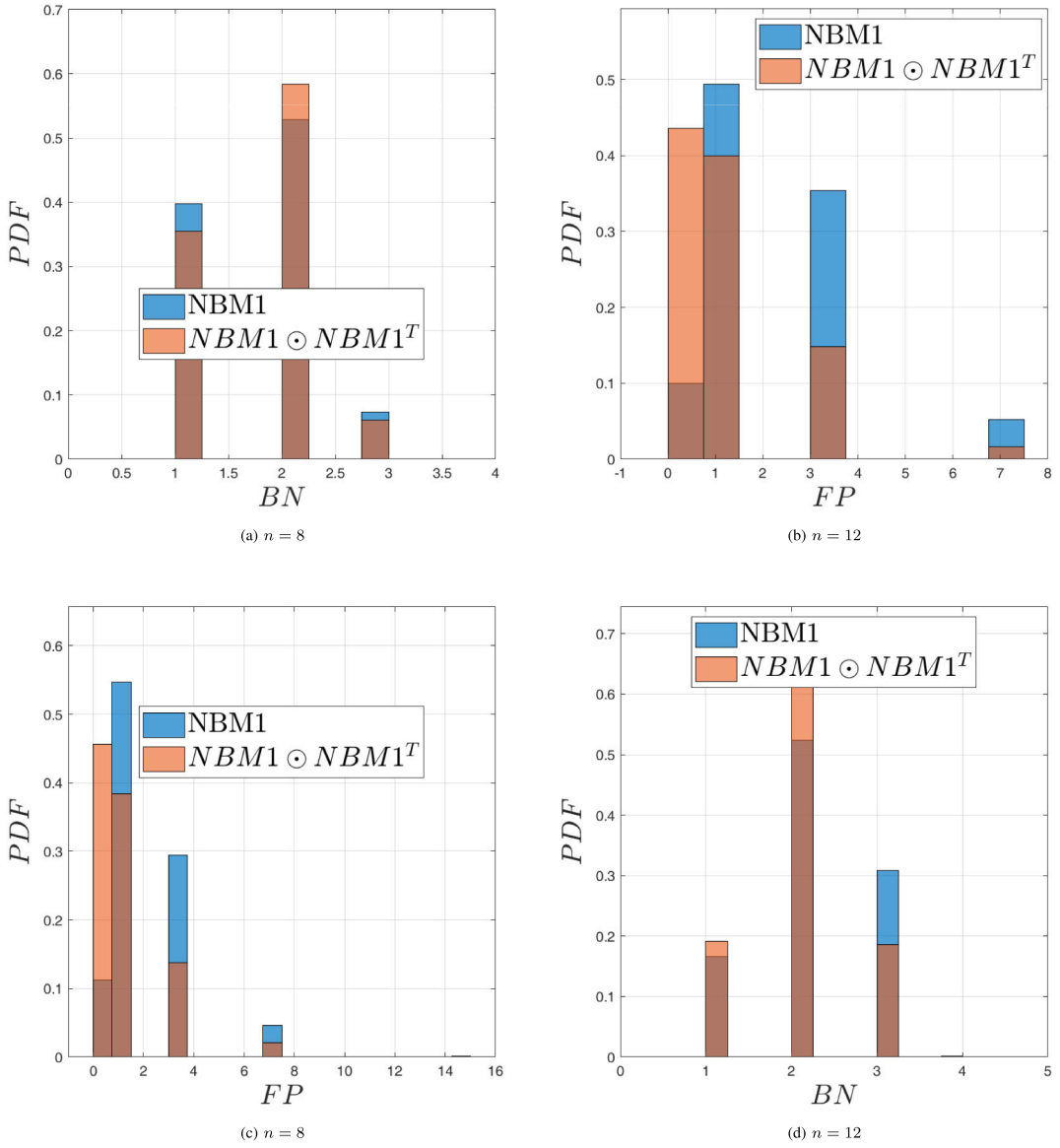


Fig. 11. Distribution of the BN (a)–(b) and FP (c)–(d) by using the proposed first non-invertible matrix forms with $n = 8$ and 12, respectively.

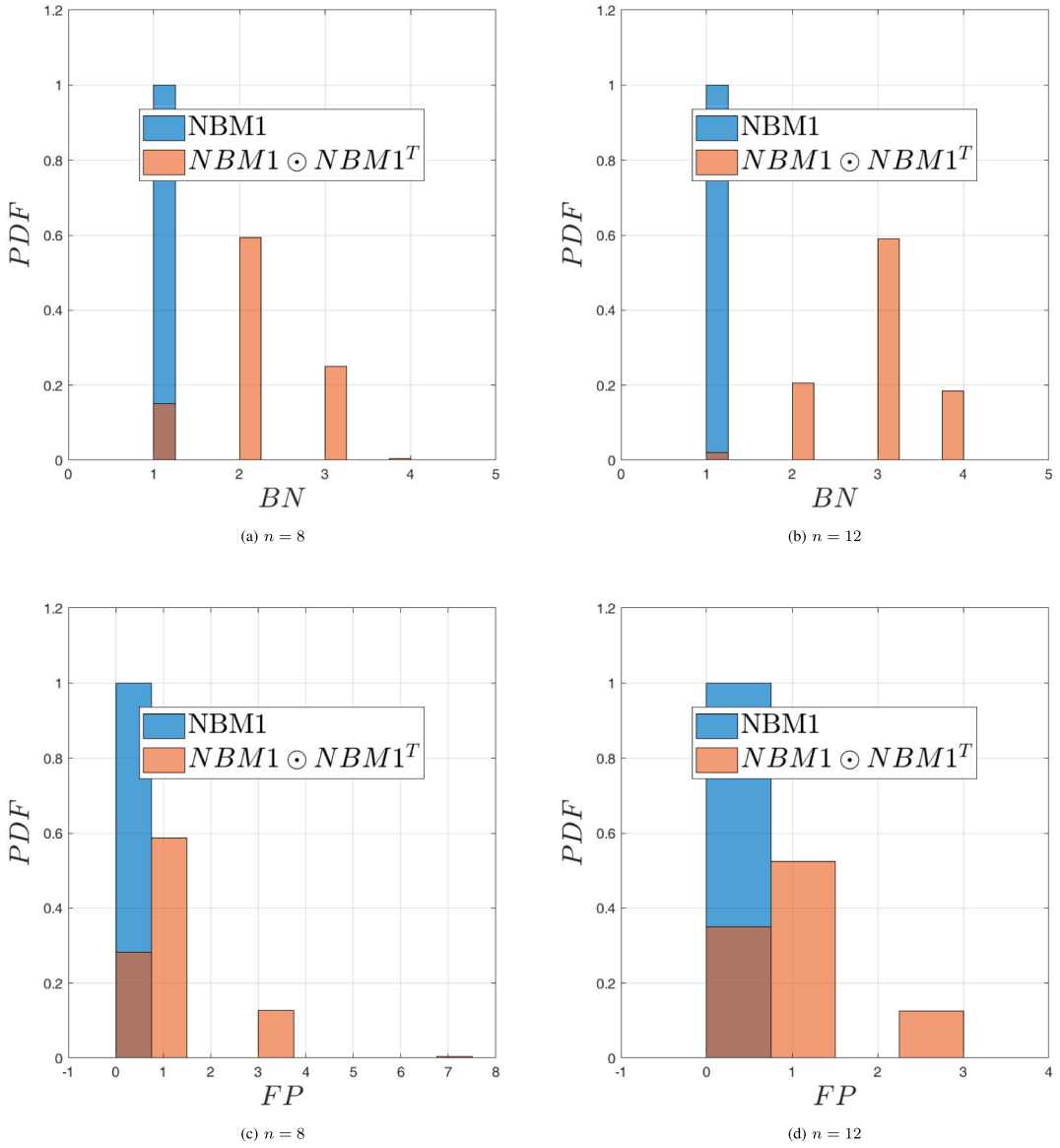
complexity, which reduces the time and resources required for their construction. The generation of a primary diffusion matrix that uses the lower or upper triangular matrix (second form) requires a keystream of size $(\frac{n^2-n}{2})$ bits. In this context, the construction of the second primary matrix form (invertible or non-invertible) is simpler compared to the other forms since it does not need to apply any additional operation such as the binary sub-matrix multiplication and/or XOR operation between two sub-matrices, each of size $(\frac{n}{2} \times \frac{n}{2})$. However, for the other primary invertible and non-invertible diffusion matrix forms, two square sub-matrices(Mu and Mv) are required. In this case, a keystream of size $(\frac{n^2}{2})$ bits is required and $\frac{n^2}{4}$ bits are used to form each square sub-matrix.

Concerning the computational complexity of the other forms: the first primary invertible diffusion matrix form requires 4 XOR sub-matrix operations in addition to one binary sub-matrix multiplication operation. The first primary non-invertible diffusion matrix form requires 2 XOR sub-matrix operations and 2 binary sub-matrix multiplication operations. The third primary invertible matrix form requires to calculate the inverse of sub-matrix Mv in addition to 3 XOR sub-matrix operations. Finally, the fourth primary invertible diffusion matrix form requires the inverse calculation of both sub-matrices (Mu, Mv), in addition to one XOR sub-matrix operation and 2 binary sub-matrix multiplication operations.

Also, the enhancement scheme requires an additional binary matrix multiplication of a dynamic primary matrix $(n \times n)$ and its transposed one (except the primary matrix form that uses the lower or upper triangular matrix that will be multiplied with a pseudo-random binary matrix). Note that the required number of operations is low in general, which confirms the efficiency of the proposed solution. Note that, to the best of our knowledge, the literature does not include similar approaches for the construction of dynamic binary diffusion primitives, which justifies the originality of the proposed solution.

5.2. Memory consumption

To produce a diffusion matrix, we need two square sub-matrices (Mu and Mv), each with $(\frac{n}{2} \times \frac{n}{2})$ bits, for all primary matrix forms that are not based on upper or lower triangular matrices. For the enhanced scheme, there is a need for a primary matrix and its transposed one, or a pseudo-random matrix. Thus, the required memory consumption to construct a diffusion matrix of $(n \times n)$ binary matrix is $(\frac{n^2}{2} + n^2)$ bits,



(a) $n = 8$ (b) $n = 12$

(c) $n = 8$ (d) $n = 12$

Fig. 12. Distribution of the BN (a) and FP (b) by using the proposed second non-invertible matrix forms with $n = 8$.

which is suitable for limited memory devices. Moreover, we can reduce the required memory consumption by reducing the value of n.

5.3. Flexibility

The importance of this work stems from the ability to produce binary diffusion matrices for any dimension n. Moreover, the required computational complexity and memory consumption are directly related to n. Thus, for constrained devices, a smaller value of n is recommended, whereas for powerful devices, a higher value of n is preferable.

5.4. Comparison with existing static schemes

As indicated previously, one of the future research directions in the design of modern symmetric cryptographic algorithms is to use dynamic key-dependent cryptographic primitives to reach better resistance against side-channel attacks and better efficiency. The efficiency is improved because the use of dynamic cryptographic primitives provides the required security level with a lower number of rounds than static ones. Moreover, the time needed for the initialization process to produce the required binary diffusion matrices is low due to the use of simple logical operations at the sub-matrix level.

In the following analysis, we evaluate the computational delay of the static standard block cipher (AES) and that of the proposed one that aims to achieve a high level of security with a minimum round number to minimize the computational complexity and associated delays and resources. The AES operations are listed below:

(1) O_S represents the byte substitution operation for a block of 16 bytes.
(2) O_D represents the AES diffusion mix-column operations (for all 4 columns and each column has 4 bytes). This operation requires the highest delay compared to the other ones.
(3) O_{xor} represents the logical XOR execution operation between two blocks of 16 bytes.
(4) O_{SR} represents the AES permutation “Shift-rows” operation of 16 bytes.
(5) O_{RKG} represents the generation of a round key operation of 16 bytes.
(6) r represents the number of rounds and it depends on the secret key size.

Table 3
Comparison between proposed dynamic binary diffusion process and existing ones.

Block cipher	Dimension	Diffusion technique	Structure	Flexibility	Invertible/Non-Invertible property
AES	4×4	MDS matrix over $GF(2^8)$	Static	No	Invertible
Camellia	8×8	Binary matrix	Static	No	Invertible
ARIA	16×16	Binary matrix	Static	No	Invertible
Proposed approach	$n \times n$	Binary matrix	Dynamic	Yes	Invertible and non-invertible

The total Number of Operations (NO) to encrypt/authenticate one block of 16 bytes using AES [29] is:

$$NO_{AES} = rO_S + (r+1)O_{xor} + (r+1)O_{RKG} + (r-1)O_D + rO_{SR} \quad (33)$$

Given that the minimum value of r in AES is 10 for a 128-bit secret key, the minimum NO of AES is given by:

$$NO_{AES(r=10)} = 10O_S + 11O_{xor} + 11O_{RKG} + 9O_D + 10O_{SR} \quad (34)$$

Concerning the proposed dynamic scheme, we show below the NO needed to process one block of 16 bytes. We consider a round number $r' < r$, varying from 2 to 6, and a binary diffusion process that uses $n = 16$:

$$NO_{prop} = (r'-1)T_{D'} + r'O_S + (r'+1)(O_{xor} + O_{RKG}) \quad (35)$$

where $O_{D'}$ represents the binary diffusion process as described in this paper, and O_{BDG} represents the generation operation of a binary diffusion matrix. Let us indicate here that O_{SR} is not required in this solution since the diffusion is done at the block level instead of the sub-block level, as in the case of AES. In addition, the number of O_{BDG} is related to the configuration (can be for one block or a set of blocks).

The objective of this discussion is to validate that in the dynamic key approach, the round number can be reduced, which decreases the overhead without degrading the security level.

In parallel, the security level as indicated previously in the paper cannot be formal when using dynamic cryptographic primitives, but a heuristic test was applied to the proposed diffusion matrix forms, and the obtained results indicate that the generated binary diffusion matrices (invertible or not) achieve on average an acceptable branch number, and lower fixed points. Also important to note that the cryptographic primitives vary and thus, they are unknown to attackers, which leads to random physical properties and hence, complicates the process of carrying analytic attacks as well as side-channel attacks [16].

Table 3 shows the difference in characteristics between the proposed dynamic binary diffusion process and the existing ones.

6. Conclusion

Towards resisting the different kinds of recent and powerful analytic and implementation attacks, the use of dynamic cryptographic primitives, such as diffusion, has been recommended recently as an alternative to the static approach. In this paper, we propose a set of binary, flexible, and dynamic diffusion (invertible or non-invertible) forms that can be employed to ensure the diffusion property. These diffusion forms exhibit simple software and hardware implementations, and they are flexible to cater for the different characteristics of utilized devices. More specifically, six diffusion matrix forms were proposed, four that are invertible and two that are non-invertible. One of the important contributions is the new construction technique for binary diffusion matrices in a pseudo-random manner, in both formats, invertible and non-invertible. However, employing only one diffusion matrix for each form does not always provide the required cryptographic performance. Therefore, for such forms, we proposed an enhanced scheme based on applying the adapted binary matrix multiplication operation between the primary matrix form and its corresponding transposed one (or with a pseudo-random matrix for the second non-invertible primary matrix form). The experimental results confirmed the enhancement of the cryptographic performance. In summary, the dynamic diffusion construction technique has the required properties for any modern cryptographic algorithm (symmetric or un-keyed), and channel encoding scheme. In addition, the generated forms are designed to strike a good balance between robustness and efficiency.

Funding

This paper is funded by the EIPHI Graduate School (contract "ANR-17-EURE-0002"), France.

Ethical approval

This article does not contain any studies with human participants or animals performed by any of the authors.

CRediT authorship contribution statement

Hassan N. Noura: Software, Writing – original draft, Writing – review & editing, Conceptualization, Methodology. **Ola Salman:** Validation, Reviewing and editing, Conceptualization, Methodology. **Ali Chehab:** Validation, Reviewing and editing, Conceptualization, Methodology.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

No data was used for the research described in the article.

References

[1] Menezes AJ, Van Oorschot PC, Vanstone SA. Handbook of applied cryptography. 1997, cited By (since 1996)5122.
[2] Stallings William. Cryptography and network security: Principles and practice. Pearson Upper Saddle River, NJ; 2017.
[3] Chou Edward, Tramèr Florian, Pellegrino Giancarlo, Boneh Dan. SentiNet: Detecting physical attacks against deep learning systems. 2018, CoRR abs/1812.00292.
[4] Schneier Bruce. Description of a new variable-length key, 64-bit block cipher (blowfish). In: Anderson Ross, editor. Fast software encryption. Lecture notes in computer science, vol. 809, Springer Berlin Heidelberg; 1994, p. 191–204.
[5] Miller Frederic P, Vandome Agnes F, McBrewster John. Advanced encryption standard. Alpha Press; 2009.
[6] Nakahara Jr J, Abrahão E. A new involutory MDS matrix for the AES. Int J Netw Secur 2009;9(2):109–16.
[7] Aoki K, Ichikawa T, Kanda M, Matsui M, Moriai S, Nakajima J, et al. Camellia: A 128-bit block cipher suitable for multiple platforms - Design and analysis. LNCS 2001;2012(2012):39–56.
[8] Kwon D, Kim J, Park S, Sung SH, Sohn Y, Song JH, et al. New block cipher: ARIA. In: Lecture notes in computer science (Including subseries lecture notes in artificial intelligence and lecture notes in bioinformatics), vol. 2971, 2004, p. 432–45, cited By (since 1996)22.
[9] Kwon D, Sung SH, Song JH, Park S. Design of block ciphers and coding theory. Trends Math 2005;8(1):13–20.

[10] Koo BonWook, Jang HwanSeok, Song JungHwan. Constructing and cryptanalysis of a 16 × 16 binary matrix as a diffusion layer. In: Chae Ki-Joon, Yung Moti, editors. Information security applications. Lecture notes in computer science, vol. 2908, Springer Berlin Heidelberg; 2004, p. 489–503.

[11] Koo BonWook, Jang HwanSeok, Song JungHwan. On constructing of a 32 ×32 binary matrix as a diffusion layer for a 256-bit block cipher. In: Rhee MinSurp, Lee Byoungcheon, editors. Information security and cryptology – ICISC 2006. Lecture notes in computer science, vol. 4296, Springer Berlin Heidelberg; 2006, p. 51–64.

[12] Aslan B, Sakalli MT. Algebraic construction of cryptographically good binary linear transformations. Secur Commun Netw 2014;7(1):53–63.

[13] Sakallı Muharrem Tolga, Akleylek Sedat, Aslan Bora, Bulus Ercan, Sakallı Fatma Büyüksaraçoglu. On the construction of 20 × 20 and 24 × 24 Binary matrices with good implementation properties for lightweight block ciphers and hash functions.

[14] Sakallı Muharrem Tolga, Akleylek Sedat, Aslan Bora, Buluş Ercan, Sakallı Fatma Büyüksaraçoğlu. On the construction of and binary matrices with good implementation properties for lightweight block ciphers and hash functions. Math Probl Eng 2014;2014.

[15] Noura Hassan, Martin Steven, Al Agha Khaldoun, Chahine Khaled. ERSS-RLNC: Efficient and robust secure scheme for random linear network coding. Comput Netw 2014;75, Part A:99–112.

[16] Noura Hassan, Courousse Damien. Method of encryption with dynamic diffusion and confusion layers. 2021, US Patent 10, 903, 978.

[17] Dai A, Kim C, Kim J. Invertibility probability of binary matrices.

[18] Noura Hassan. Design and simulation of efficient chaos based generators, crypto-systems and hash functions [Ph.D. thesis].

[19] Noura Hassan, Martin Steven, Al Agha Khaldoun. E3SN - efficient security scheme for sensor networks. In: SECRYPT - 10th international conference on security and cryptography. Reykjavik, Iceland; 2013.

[20] Noura Hassan N, Salman Ola, Couturier Raphaël, Chehab Ali. LoRCA: Lightweight round block and stream cipher algorithms for IoV systems. Veh Commun 2021;100416.

[21] Noura Hassan N, Noura Mohamad, Chehab Ali, Mansour Mohammad M, Couturier Raphaël. Efficient and secure cipher scheme for multimedia contents. Multimedia Tools Appl 2019;78:1–30.

[22] Noura Hassan, Chehab Ali, Noura Mohamad, Couturier Raphaël, Mansour Mohammad M. Lightweight, dynamic and efficient image encryption scheme. Multimedia Tools Appl 2018;1–35.

[23] Noura Hassan, Sleem Lama, Noura Mohamad, Mansour Mohammad M, Chehab Ali, Couturier Raphaël. A new efficient lightweight and secure image cipher scheme. Multimedia Tools Appl 2017;1–28.

[24] Noura Hassan N, Chehab Ali, Couturier Raphael. Efficient & secure cipher scheme with dynamic key-dependent mode of operation. Signal Process, Image Commun 2019;78:448–64.

[25] Daemen Joan, Rijmen Vincent. The design of Rijndael. Secaucus, NJ, USA: SpringerVerlag New York, Inc.; 2002.

[26] Z'aba Muhammad Reza. Analysis of linear relationships in block ciphers [Ph.D. thesis], Queensland University of Technology; 2010.

[27] Bilmes J, Asanovic K, Chin C, Demmel J. Optimizing matrix multiply using PHiPAC: A portable, high-performance, Ansi C coding methodology. In: International conference on supercomputing. 1997.

[28] Brent RP. Algorithms for matrix multiplication. Technical report TR-CS-70-157, Stanford University; 1970.

[29] Daemen Joan, Rijmen Vincent. The design of Rijndael: AES-the advanced encryption standard. Springer Science & Business Media; 2013.