UDC 004.056.55

https://doi.org/10.55452/1998-6688-2025-22-4-131-142

[1*]Gorlov L.V.,
PhD student, ORCID 0000-0002-8208-4716,
e-mail: lev.gorlov@gmail.com
[2]Seilova N.A.,
Cand. Tech. Sc., Associate Professor, ORCID: 0000-0003-3827-179X,
e-mail: nseilova@iitu.edu.kz
[3]Okhrimenko T.A.,
Cand. Tech. Sc., Associate Professor, ORCID: 0000-0001-9036-6556,
e-mail: t.okhrimenko@nau.edu.ua.

[1]Al-Farabi Kazakh National University, Almaty, Republic of Kazakhstan
[2]International University of Information Technologies, Almaty, Republic of Kazakhstan
[3]National Aviation University, Kiev, Ukraine

ON THE DIFFUSION LAYER GENERATION METHOD

Abstract

This paper presents an automated method for generating the parameters of linear functions used in the diffusion layer of block symmetric encryption algorithms. The focus is on designing linear layers constructed solely from cyclic shift operations and bitwise XORs, which are both efficient and hardware-friendly. Such layers play a critical role in achieving strong diffusion, a fundamental cryptographic requirement. The proposed method evaluates candidate configurations by exhaustively enumerating shift values, calculating their branch number, and assessing their avalanche characteristics. A set of quantitative diffusion metrics is introduced to guide the selection process, including single- and multi-round avalanche effects and activation rates at the byte level. An aggregated quality function is formulated to allow comparative assessment. The developed software tool identified optimal shift parameters for 128-bit blocks processed as four 32-bit words, achieving a branch number of 5 with only 12 XOR operations. The proposed approach contributes to the practical synthesis of lightweight and secure cryptographic primitives suitable for both classical and constrained platforms.

Keywords: Block symmetric encryption algorithm, linear layer, branch number, avalanche effect.

Introduction

The purpose of this paper is to show methods for generating shift coefficients for the linear layer of the architecture presented in "Linear Layer Architecture Based on Cyclic Shift and XOR" as well as optimization methods that reduce the effort required to enumerate candidate coefficients by orders of magnitude. Preliminary conclusions about cryptographic qualities and computational complexity were given earlier in [1]. General approaches to balancing the function – such as adjusting the number and length of words and enhancing it with additional iterations – were also discussed, and a comparison with the AES algorithm's linear layer was provided.

Evaluation of cryptographic resistance against linear and differential attacks is also beyond the scope of this paper and will be published in the future.

Modern block encryption algorithms are built on the principles formulated by Claude Shannon: the principle of confusion and diffusion [2, 3]. Diffusion is achieved, in particular, by using linear transformations with good diffusion characteristics, which ensure rapid propagation of changes across all bits of the output vector.

One of the key metrics used to evaluate the diffusion ability of a linear function is the branch number. This indicator reflects the minimum possible number of different words at the input and output of a linear transformation:

$$B(L) = min_{x \neq 0}\{w(x) + w(L(x))\} \quad (1)$$

where w is Hamming weight. The sizes of words and their location within the converted data block may have different values depending on the specific architecture and operations used in the encryption algorithm. As a general case we will consider words as the input of the nonlinear function of the algorithm, the size of which is typically equal to 8 bits.

In both classical and lightweight encryption algorithms, the linear diffusion layer is often the most computationally intensive component [4–6]. Linear operations based on maximum distance codes (MDS) are widely used to perform linear transformations in various encryption algorithms, including AES due to the strictly proven branch number [7–11]. However, one can consider alternative functions that may not have the largest possible branch number, but have sufficiently strong cryptographic properties with significantly higher performance [12, 13].

Schemes using cyclic shift and XOR on words of different lengths are already used in lightweight encryption algorithms to mix the bits of the algorithm's internal state [14–16]. These transformations do not require reversibility and can include additional operations. For example, the ARX (Add-Rotate-XOR) architecture includes modular addition modulo 2^y, where y represents the word length.

There are also algorithms for constructing transformations equivalent to multiplication by MDR matrices using cyclic shift and bitwise addition operations. The papers provide a theoretical basis for obtaining such operations analytically, without exhaustive search, for a block of four words [17–19]. The linear operation constructed in this way has a branch number of five, which is the maximum achievable for an operation involving four subblocks.

At present, no information has been found on the possibility of generating larger matrices with a higher branch number using the same method.

Materials and methods

The goal is to construct a linear function that facilitates rapid and uniform propagation of input changes across all output bits, thereby preventing the construction of effective differential or linear attacks [2] . The main objective of this paper is to formalize the approach to assessing the quality of a cipher's linear layer and to propose universal criteria that allow quantitative comparison of various designs in order to generate a linear function with characteristics close to optimal.

Since the calculation of the branch number is a computationally complex task and cannot be used as a criterion in enumeration problems, for the initial assessment of the efficiency of the linear function $L: \mathbb{F}_2^n \to \mathbb{F}_2^n$ we propose to consider a set of five complementary characteristics:

1. Average avalanche effect per round (denoted as criterion A_1) is the average number of bits at the output of the function that change when one bit at the input is inverted. This metric characterizes local diffusion and shows how quickly the initial change spreads along the output vector. In the ideal case, $A_1 \approx n/2$, but for a lightweight function this result is hardly achievable.

Average number of bytes activated per round (B_1). Given the influence of the nonlinear layer operating on bytes it is reasonable to consider how many bytes will change in each round of the algorithm when at least one bit of the input changes. This parameter is most closely related to the branch number used in the wide trail design strategy [3].

Average avalanche effect over two rounds (A_2) – similar to criterion A_1, but measured over two consecutive rounds (ignoring the nonlinear layer between those rounds). This characteristic better reflects the cumulative diffusion effect during the operation of the encryption algorithm.

Average number of activated bytes over two rounds (B_2).

Minimum number of output bits activated by a single active input bit (B_{min}):

$$B_{min} = \min_{x:w(x)=1} w\big(L(x)\big) \tag{2}$$

This parameter defines the worst-case diffusion scenario and is also indirectly related to the branch number. Its increase directly contributes to increasing the cipher's resistance to differential and linear cryptanalysis.

For a systematic assessment of the linear function based on the above parameters, it is proposed to use an aggregated metric in the form of a weighted criterion:

$$Score(L) = \alpha \cdot \frac{A_1}{n} + \beta \cdot \frac{A_2}{n} + \gamma \cdot \frac{B_1}{n} + \delta \cdot \frac{B_2}{n} + \theta \cdot \frac{B_{min}}{n} \tag{3}$$

here $\alpha, \beta, \gamma, \delta, \theta \in \mathbb{R}_{>0}$ – normalized weighting factors determined by design priorities. In this case, all values are normalized by n to bring them to a single scale. For example, when focusing on cryptographic resistance, one can choose $\gamma > \alpha, \beta$, while in problems of optimizing the diffusion rate in a small number of rounds, one can increase the significance of β.

Thus, the formulated criterion allows one to compare various linear functions taking into account both average and extreme behavior, and can be used as a heuristic in the automated synthesis of diffusion layers.

It is required to generate a linear function that is implemented by computationally easy operations and has a branch number of at least 5 for a data block of 16 words of 8-bit size [21–24]. These values are taken by analogy with the AES algorithm and other common encryption algorithms and are de facto standard for block symmetric encryption algorithms [25–30]. The following considerations were taken into account in its development:

A given branch number can be achieved by performing an operation on at least 4 words. A 128-bit data block is extremely conveniently divided into 4 words of 32 bits, since most modern computing platforms implement operations on 32-bit words in an optimal way [4].

To ensure mutual influence of bits located at different positions, a cyclic shift operation is necessary, and to change the values of bits, the XOR operation is optimal.

For inversion to be possible, the resulting linear operation must be bijective.

The influence of words should be mutual, so at each iteration either one word should overlap all the others, or vice versa, which seems easier to implement.

Taking these requirements into account, a linear operation over an arbitrary number of words of arbitrary (identical) length is constructed, allowing for balancing diffusion properties and computational complexity [1].

The resulting linear function f consists of several successive operations g over a set of words $a = \{a_0, \dots, a_{u-1}\}$ and is expressed by the following formula:

$$g_{(j)}: a^{(n+1)} = g_{(j)}\big(a^{(n)}\big) \Leftrightarrow a_k^{(n+1)} = \begin{cases} \bigoplus_{i=0}^{u-1} \big(a_{i+j \bmod u}^{(n)} \lll C_i\big), k = j \\ a_k^{(n)}, k \neq j \end{cases} \tag{4}$$

The operation f, which changes all the words of the processed block, can be represented as follows:

$$f(a) = g_{(0)}(a) \circ \ldots \circ g_{(u-1)}(a) \quad (5)$$

The value of the shift constants $C_{0\ldots u-1}$ is of decisive importance for the efficiency of the function. Clearly, not all sets of values lead to the desired distribution of mutual influence of bits.

Proving that the branch number of a function is 5 is a computationally difficult problem [32–34]. For all possible inputs of a linear function, it is necessary to enumerate all combinations of $d = 1 \ldots 4$ active bytes and make sure that at least $5 - d$ bytes are activated at the output of the function. There are 2^{128} inputs to the linear function, each of which would require iterating over $2^8 \times 16$ variants of the difference in the input in a single byte, for which the difference in the output must be at least 4 bytes, $2^{16} \times 16 \times 15$ two-byte input differences, and so on, for a total of $\sum_{d=1}^{4} 2^{8d} \times A_d^{16} \approx 2^{48}$ operations. Thus, solving the problem as a whole using brute force requires about 2^{176} operations, which is practically impossible.

However, since the function is linear, the influence of each input bit on the output bits can be found independently of the other bits. This property allows for a preliminary estimate of the function coefficients that is not directly related to the branch number and avalanche effect, but is correlated with them to some extent.

The advantage of this method for estimating the shift coefficients of a linear function is its low computational complexity (namely, 2^7 linear for one set of coefficients and 2^{22} for the algorithm as a whole without taking into account further optimizations) and, as a consequence, the possibility of a complete enumeration of the entire set of coefficients.

The main drawback is the lack of a specific resulting value of the branch number. The avalanche effect for some combination of input bits may also be less than the number of output bits activated by a single non-zero input bit due to the peculiarity of the XOR function: some activated bits may overlap each other and ultimately have no effect on the output. Calculating all possible combinations of mutual overlaps to estimate the branch number has the same computational complexity as the full search above, namely 2^{176} operations.

However, it is possible to optimize both the evaluation of the primary properties of the function and the verification of the branch number of the resulting function.

First, enumerating all input values of a linear function does not make sense, since the changed input bits will affect the same output bits regardless of the values of the remaining (fixed) input bits. This allows for direct enumeration of differences, abstracting from enumerating all input combinations. This reduces the enumeration by 2^{128} times.

The next property that allows optimizing the calculation of the lower bound of the branch number of a linear function is its cyclicity. In particular, due to the cyclic structure of the transformation, different input vectors obtained by cyclically shifting each other will yield identical (with the same relative shift) output differences. Thus, the set of all configurations with 4 active blocks is divided into orbits by the action of a group of cyclic shifts, and it is sufficient to consider only one representative from each orbit. Since in the case under consideration not only the number of active bits is important, but also the number of active bytes, it makes sense to exclude from the search input blocks that are equivalent in value and shifted relative to each other by a number of bits multiple of 8. This significantly reduces the volume of the search and allows focusing on unique structures of input differences from the point of view of the output. This approach is consistent with the general principles of analyzing the diffusion properties of linear transformations and is widely used in assessing the differential characteristics of cryptographic primitives with a regular architecture [35–36].

It should be noted that this equivalence is not preserved during a cyclic shift of the block, i.e. when transferring the most significant bits of one word to the positions of the least significant ones of another word, since the influence of the bits, propagated by a linear operation, depends on the word in which they are located in the block.

In practice, this allows us not to go through all possible options for placing active bytes. It is sufficient to consider only those in which the least significant byte of each row is active. For example, let us consider two input blocks that give outputs equivalent to the shifts:

		B	A
	D	C	

	B	A	
D	C		

Figure 1 – Inputs of a linear function equivalent by branching

Here the values of the remaining bytes are 0, therefore they have no effect on the activation of the output bits.

This optimization allows us to strictly calculate the lower bound of the branch number in $\sum_{b=1}^{4} 2^{8b} A_{b-1}^{16-b} \approx 2^{43}$ operations. There are certainly methods that can reduce the search even further, for example, by eliminating other subsets of bits through other types of equivalence, but with the current approach, the computation can be completed within a few minutes of computing on a modern processor.

The same approach can be applied to the evaluation of the primary indicators of a function in order to reduce the search space for single active-bit inputs. Indeed, all output options for different bits of one input word are cyclic shifts of each other. And similarly, bits of different input words generate different combinations of outputs. In this case, it is necessary to estimate the number of both active bits and active bytes at the output of the function, therefore, to fully find the values of interest, it is necessary to sort through 32 input options (all single bits of one of the bytes for each word).

Results and discussion

The desired linear transformation is constructed using bitwise cyclic shifts and XOR operations on four 32-bit words and has the form (in the C programming language):

dout[0] = din[0] ^ ROTL(din[1], c[0]) ^ ROTL(din[2], c[1]) ^ ROTL(din[3], c[2]);

dout[1] = din[1] ^ ROTL(din[2], c[0]) ^ ROTL(din[3], c[1]) ^ ROTL(dout[0], c[2]);

dout[2] = din[2] ^ ROTL(din[3], c[0]) ^ ROTL(dout[0], c[1]) ^ ROTL(dout[1], c[2]);

dout[3] = din[3] ^ ROTL(dout[0], c[0]) ^ ROTL(dout[1], c[1]) ^ ROTL(dout[2], c[2]);

To find the coefficients with the desired properties, it is necessary to perform a complete enumeration of all possible triplets of shift values $c_i \in [0,31], i = \{1,2,3\}$.

For each triple of coefficients, the average diffusion effect and the minimum number of activated output bytes must be determined over all possible single-bit input differences. The avalanche effect of a lightweight linear function extends to the entire processed block in several rounds, therefore, when selecting the coefficients of the functions, the avalanche effect was also evaluated over two rounds (i.e., the linear layer was applied twice). In practice, when using sufficiently high-quality nonlinear blocks this effect is significantly amplified, since when changing at least one bit of the output of a linear function, an average of 4 bits related to the same substitution block will be changed [37].

For each tested set of coefficients:

1) All possible inputs with a single one bit (the other bits are 0) are iterated over. Thus, all the following steps are performed 32 times for each combination of shift values;
2) The block is fed to the input of the doubled linear function;
3) The number of nonzero bytes at the output is counted;
4) Results for any coefficient set with fewer nonzero output bytes than the current best are discarded;

5) The average value of the number of activated bits is found. If it is greater than the best current result, then the set of constants is accepted as the best current result.

Point 4 of the algorithm allows significant optimization of the search, since at the early stages the values that give bad special cases are filtered out.

Practical selection gave many results with similar characteristics. In particular, 7217 sets of coefficients were found, in which any single active input bit activates at least 6 output bits.

To select the coefficients of a linear function with the corresponding cryptographic criteria, it is necessary to select the weight coefficients $\alpha, \beta, \gamma, \delta, \theta$ from formula (3).

The choice of these coefficients logically follows from the significance of the corresponding criteria imposed on the linear function being developed, which is beyond the scope of this paper. Below, some initial provisions for selecting coefficients without strict justifications will be presented.

Apparently, the coefficient θ does not play a big role, since it is always advisable to choose a function that activates the maximum number of bits with a single active input bit, i.e. to consider only the coefficients that generate functions with B_{min} equal to 7.

In the presence of a nonlinear block with a powerful avalanche effect, the weights β and δ can be significantly reduced or even set to zero.

For an algorithm with a large number of rounds, on the contrary, it makes sense to rely on two-round indicators, reducing α and γ.

The results of the program implementing this algorithm with different weighting coefficients are presented below:

Table 1 – Examples of generation results

№	α	β	γ	δ	θ	A_1	A_2	B_1	B_2	B_{min}	c_0	c_1	c_2	Score
1	1	0	1	0	1	8.75	15.75	8.12	12.12	7	1	10	15	7.958
2	0	1	0	1	1	8.75	22.25	7.56	13.88	7	1	17	14	14.375
3	1	0.5	2	0.8	1	8.75	21.75	7.97	14.12	7	2	26	25	10.163
4	1	2	1	2	1	8.75	22.25	7.66	13.88	7	1	18	14	13.665

To check the functionality of the function that calculates the branch number of a function of the form (5), a test was conducted in which an optimized enumeration method was used to search for input values of up to 4 active bytes that yield the minimum output weight for a function variant with shift coefficients {1, 10, 15}.

The search was performed in parallel in several threads covering subranges of input byte values. To analyze all possible values of three of the four bytes for all possible values of the first byte of the first word, four parallel instances of the software were run, resulting in the following worst-case outputs:

Table 2 – Worst cases for first word byte enumeration

№	Diapason	BI	Input/output (hexadecimal)
1	0..63	5	00 09 00 00 80 00 00 00 40 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 40 00 00 00 00 00 01 00
2	64..127	5	48 00 00 00 04 00 00 00 02 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 02 00 00 00 00 08 00 00
3	128..191	5	90 00 00 00 08 00 00 00 04 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 04 00 00 00 00 10 00 00
4	192..255	5	d8 00 00 00 0c 00 00 00 06 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 06 00 00 00 00 18 00 00

For an input difference in the first byte of the second word, the outcome was even better, since

the second word's bits influence the first word from the very first iteration of the linear function:

Table 3 – Results of enumeration for bytes of the second word

№	Diapason	BI	Input/output (hexadecimal)
1	0..127	7	00 00 00 00 00 00 00 00 10 00 00 00 00 00 00 80 00 00 00 00 30 00 00 00 11 c0 00 00 00 42 00 83
2	127..255	5	00 00 00 00 80 00 00 00 08 00 00 00 00 00 00 00 00 00 00 00 90 00 00 00 08 40 02 00 00 32 00 09

For the third and fourth words, no cases were found in which less than 7 bytes were activated in two adjacent rounds.

Thus, the minimum sum of the input and output weights of a one-round linear function is 5, and therefore the branch number of a linear function of the form (5) with coefficients {1,10,15} is 5.

Comparative analysis of linear layer design

Architecture. Traditional diffusion layers in block ciphers often use matrices from maximum distance separable (MDS) codes (e.g., the MixColumns matrix in AES) to achieve linear mixing [5]. These MDS-based designs perform linear transformations (usually over finite fields) with a strictly proven diffusion property (maximal branch number) [3]. In contrast, the proposed method uses only cyclic bit shifts and XOR operations – a shift/XOR–based linear layer similar to ARX (Add-Rotate-XOR) constructions [6]. This means that instead of heavy finite-field multiplications or complex linear combinations, diffusion is achieved through simple rotations and XORs. Such an approach is hardware-friendly and simplifies implementation, while still maintaining the linear invertibility required for cipher rounds. The shift/XOR design aligns with many modern lightweight ciphers, which favor simple operations for efficiency, distinguishing our method as a purely bitwise-linear alternative to classical MDS matrices.

Branch number. MDS-based layers are valued because they guarantee the maximum possible branch number for a given matrix dimension [3]. For example, an $n \times n$ MDS matrix ensures the highest minimum number of active input/output words (the branch number) by design. Shift/XOR-based layers generally do not inherently guarantee this theoretical maximum; they often achieve near-maximum diffusion in practice. However, a key result of this work is that through careful parameter selection we can attain a branch number equal to the MDS optimum. In fact, for a block of four 32-bit words (128-bit state), our optimized rotation–XOR construction reaches a branch number of 5, which is the provable maximum for any 4-word linear transformation [7]. This demonstrates that even without using an MDS code, the proposed method can match the diffusion strength of an MDS matrix in this scenario. While classical MDS matrices provide a mathematical proof of maximum branch number, our approach shows that near-MDS or even MDS-equivalent branch numbers are achievable with a shift/XOR design – combining strong diffusion with the benefits of a lightweight operation. This result underscores the novel contribution of attaining high branch number without resorting to traditional MDS constructs.

Designing complexity. Designing an MDS-based linear layer often leverages algebraic methods or known constructions from coding theory [7]. In general, finding an MDS matrix for a given size can be done via mathematical criteria or combinatorial search, but many such matrices are known or can be constructed using finite-field techniques. By contrast, identifying an optimal shift/XOR configuration required brute-force search over rotation constants. The search space is enormous – yet our method introduces a highly optimized enumeration strategy to make this design problem tractable. We exhaustively evaluate candidate shift combinations, but we guide the search with quantitative diffusion metrics and pruning heuristics. Notably, we reduce the complexity of the search by several orders of magnitude through intelligent heuristics and early elimination of suboptimal candidates. This

optimized search procedure (with metrics like single- and two-round avalanche criteria) effectively navigates the design space, in contrast to a naive brute force. The result is a practical computational method that finds excellent diffusion layers (even matching MDS-level branch number) without an explicit algebraic formula. Thus, whereas MDS-based designs benefit from direct theoretical constructions, the proposed shift/XOR design required developing an automated search framework – an effort justified by the significant performance gains of the final design.

Implementation complexity. A well-known drawback of many MDS-based layers is their implementation cost. Linear diffusion via an MDS matrix (especially over bytes) often becomes the most computationally intensive component of the cipher. Implementing finite-field multiplications or look-up tables for an MDS matrix can incur a high gate count, many XOR operations, or additional memory for tables [39–40]. In contrast, the shift/XOR approach uses only primitive CPU operations (bit rotates and XORs), leading to significantly lower complexity in both hardware and software. The proposed linear layer achieves its diffusion with a remarkably small number of operations – for a 128-bit block we use only 12 XOR operations in total (along with shifts) to attain branch number 5. There are no multiplications or large tables needed, which translates to fewer logic gates and faster execution on microcontrollers. This efficiency makes the method highly suitable for lightweight and constrained devices, where minimizing computational overhead is crucial. In summary, while an MDS-based layer might maximize diffusion at the cost of more intensive computation (often requiring careful optimization or special implementation tricks), the shift/XOR-based layer offers a far more lightweight solution. The novel contribution here is demonstrating equivalent diffusion strength with drastically reduced operation count, thereby improving encryption performance on resource-limited platforms without sacrificing security.

Flexibility and Adaptability. MDS code-based layers are typically tied to specific block and word sizes – for example, AES's 4×4 byte matrix is fixed to 128-bit blocks (16 bytes), and designing a new MDS matrix for a larger state or different word size can be non-trivial. Each change in dimensions often requires finding new matrices that preserve the MDS property, a process that can become a research problem in itself for higher dimensions. In contrast, the proposed shift/XOR generation method is inherently more flexible. Our framework can be extended to an arbitrary number of words and word lengths by selecting appropriate rotation coefficients. In principle, the same approach can synthesize a diffusion layer for different block sizes (e.g., 64-bit blocks or 256-bit blocks) by adjusting the word partitioning and rerunning the automated search for optimal shifts. This scalability is enabled by the general nature of rotations and XORs, which are not locked to byte-oriented operations or a fixed field size. Furthermore, the method is adaptable in terms of design goals: we can tune the weightings of various diffusion metrics to prioritize certain criteria. For instance, if the cipher uses a very strong nonlinear layer (S-box), the search can de-emphasize single-round diffusion and focus on multi-round effects, or vice versa. The authors explicitly allow adjusting parameters (like weighting factors for avalanche criteria) to suit different scenarios – e.g., ignoring some diffusion criteria when an S-box already provides good mixing, or emphasizing two-round diffusion for ciphers with many rounds. This level of adaptability is a distinct advantage of the proposed approach. In summary, unlike a one-size-fits-all MDS matrix, our shift/XOR-based generation method can scale and adapt to various block architectures and security requirements with relative ease. This flexibility in accommodating different word sizes and balancing performance vs. diffusion trade-offs highlights the novel contribution of our method in the landscape of linear layer design.

Conclusion

The study introduces a scientifically novel and practically applicable method for synthesizing high-quality linear diffusion layers without resorting to pre-designed MDS codes. The originality of the approach lies in the integration of an optimized exhaustive search with mathematically grounded

pruning techniques, which drastically reduce the search space while preserving the ability to obtain provable branch numbers. This makes it possible, for the first time in the context of purely shift- and XOR-based constructions, to confirm by direct computation the achievement of the theoretical maximum branch number of 5 for a 4-word, 32-bit architecture.

The authors' personal contributions include formulating a set of quantitative diffusion metrics that combine average-case and worst-case behavior, defining a unified weighted evaluation criterion, and implementing a specialized software tool that automates both coefficient generation and branch number verification. Additionally, the authors developed and validated new equivalence-class-based optimization strategies that reduce the combinatorial complexity of branch number checking from impractical brute-force scales to a range feasible on standard modern processors.

This work not only delivers a concrete high-performance linear transformation meeting stringent cryptographic criteria but also provides a reusable methodology for designing diffusion layers adaptable to diverse block sizes, word lengths, and implementation constraints. The proposed generation framework can serve as a foundation for further research on lightweight cipher design, particularly in scenarios where hardware simplicity, provable diffusion strength, and adaptability are equally critical.

REFERENCES

1 Gorlov, L. Iavich, M. and Bocu, R. Linear Layer Architecture Based on Cyclic Shift and XOR. Symmetry, 15, (2023).

2 Debranjan Pal, Vishal Pankaj Chandratreya, Abhijit Das, Dipanwita Roy Chowdhury. Modeling Linear and Non-linear Layers: An MILP Approach Towards Finding Differential and Impossible Differential Propagations, arXiv:2405.00441 (2024).

3 Daemen, J. and Rijmen, V. The wide trail design strategy. In Proceedings of the Cryptography and Coding: 8th IMA International, 2001, pp. 222–238.

4 Wikipedia. Word (computer architecture). URL: https://en.wikipedia.org/wiki/Word_(computer_architecture).

5 Daemen, J. and Rijmen, V. The Design of Rijndael. Springer, 2002.

6 Nir, Y. and Langley, A. ChaCha20 and Poly1305 for IETF Protocols. RFC 7539, (2018). URL: https://datatracker.ietf.org/doc/html/rfc7539. [Accessed 9.05. 2025].

7 Joshi, D. A Note on Upper Bounds for Minimum Distance Codes. Information and Control, 1(3), p. 289–295 (1958).

8 Shannon, C. Mathematical Theory of Cryptography, Bell Labs (1945).

9 Shannon, C. Communication Theory of Secrecy Systems, Bell System Tech. J. (1949).

10 Wu, S., Wang, M., and Wu, W. Design of lightweight linear diffusion layers from near-MDS matrices. International Association for Cryptologic Research. URL: https://eprint.iacr.org/2017/195.pdf. [Accessed 9.05.2025].

11 Dinu, D., Perrin, L., Udovenko, A., Velichkov, V., Großschädl, J., and Biryukov, A. Design Strategies for ARX with Provable Bounds: SPARX and LAX. Advances in Cryptology – ASIACRYPT 2016, pp. 484–513 (2016).

12 Biham, E. and Shamir, A. Differential Cryptanalysis of DES-like Cryptosystems. J. Cryptology (1991).

13 Guo, Z., Liu, R., Wu, W. and Lin, D. Direct Construction of Lightweight Rotational-XOR MDS Diffusion Layers, International Association for Cryptologic Research, 2016. URL: https://ia.cr/2016/1036. [Accessed 9. 05.2025].

14 Guo, Z., Liu, R., Gao, S., Wu, W. and Lin, D. Direct Construction of Optimal Rotational-XOR Diffusion Primitives, p. 169–187 (2017).

15 Ray, B., Douglas, S., Jason, S., Stefan, T.C., Bryan, W. and Louis, W. The SIMON and SPECK Families of Lightweight Block Ciphers. Cryptol. ePrint Arch., 2013. [Online]. URL: https://ia.cr/2013/404. [Accessed 9. 05.2025].

16 FIPS. Federal Information Processing Standards Publication 197. Specification for the Advanced Encryption Standard (AES) (2001). URL: http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf. [Accessed 9.05.2025].

17 Matsui, M. Linear Cryptanalysis Method for DES Cipher., EUROCRYPT (1993).

18 Nyberg, K. Perfect Nonlinear S-Boxes., EUROCRYPT (1991).

19 Schneier, B. Applied Cryptography, Wiley (1996).

20 Vaudenay., S. On the Need for Multipermutations., FSE (2001).

21 Panasenko, S. Algoritmy shifrovanija, BHV-Peterburg (2009).

22 Derkach, A. Perevirka model'nih pripushhen' u kriptoanalizi ARX-shifriv, Visnik NTUU «KPI». Serija: Matematichne modeljuvannja v dizajni ta ekonomici, 2, 231–236 (2020).

23 Tian, Yu. & Feng, Xiutao & Li, Guangrong. On the construction of ultra-light MDS matrices, 10.48550/arXiv.2409.03298, 2024.

24 Gupta, Kishan & Kumar Pandey, Sumit & Samanta, Susanta. On the Construction of Near-MDS Matrices, 10.48550/arXiv.2306.12791, 2023.

25 Al-Nofaie, S.M., Sharaf, S., & Molla, R. Design Trends and Comparative Analysis of Lightweight Block Ciphers for IoTs, Applied Sciences, 15(14), 7740 (2025).

26 Konstantopoulou, Evangelia and Athanasiou, George and Sklavos, Nicolas. Review and Analysis of FPGA and ASIC Implementations of NIST Lightweight Cryptography Finalists, ACM Comput. Surv., 57, 10 (2025).

27 Shi Wang, Yuan Chen, Yunqing Li, Xiangyong Zeng. On construction of lightweight MDS matrices, Advances in Mathematics of Communications, 16(4) (2022).

28 Gupta, Kishan & Kumar Pandey, Sumit & Samanta, Susanta. On the Direct Construction of MDS and Near-MDS Matrices, 10.48550/arXiv.2306.12848, 2023.

29 Yogesh Kumar, Prasanna Raghaw Mishra, Susanta Samanta, Kishan Chand Gupta, Atul Gaur. Construction of all MDS and involutory MDS matrices Advances in Mathematics of Communications, 19(3), 922–941 (2025).

30 Samanta, S. On the counting of involutory MDS matrices. Cryptography and Communications, 2024.

31 Gaëtan Leurent and Clara Pernot. Design of a Linear Layer Optimised for Bitsliced 32-bit Implementation. Cryptology ePrint Archive, Paper 2023/1803, 2023.

32 Rishakani, A.M. Cryptographic Properties of Cyclic Binary Matrices. Advances in Mathematics of Communications, 2019.

33 Luong, Tran & Long, Nguyen & Vo, Bay. Efficient implementation of the linear layer of block ciphers with large MDS matrices based on a new lookup table technique, PLOS ONE, 19 (2024).

34 Dobraunig, Christoph & Eichlseder, Maria & Mendel, Florian & Schläffer, Martin, Ascon v1.2: Lightweight Authenticated Encryption and Hashing. Journal of Cryptology, 34 (2021).

35 Leurent, G., & Pernot, C. Design of a Linear Layer Optimised for Bitsliced 32-bit Implementation, IACR Transactions on Symmetric Cryptology, 1, 441–458 (2024).

36 Mishra, P.R., Kumar, Y., Samanta, S., Gaur, A. A New Algorithm for Computing Branch Number of Non-Singular Matrices Over Finite Fields, Security and Cryptography for Networks. SCN 2024. Lecture Notes in Computer Science, 14974 (2024).

37 Huck Bennett, Mahdi Cheraghchi, Venkatesan Guruswami & Jo˜ao Ribeiro. Parameterized Inapproximability of the Minimum Distance Problem over all Fields and the Shortest Vector Problem in all ℓp Norms, SIAM Journal on Computing, 53 (2024).

38 Vijay Bhattiprolu, Venkatesan Guruswami, Xuandi Ren, PCP-free APX-Hardness of Nearest Codeword and Minimum Distance, arXiv:2503, 11131 (2025).

39 Tian, Y., Feng, X., & Li, G. On the construction of ultra-light MDS matrices. arXiv preprint (2024).

40 Kurt Pehlivanoğlu, Meltem & Demir, Mehmet Ali. Optimizing implementations of linear layers using two and higher input XOR gates. PeerJ Computer Science, 10 (2024).

[1]*Горлов Л.В.,
докторант, ORCID 0000-0002-8208-4716,
*e-mail:lev.gorlov@gmail.com
[2]Сейлова Н.А.,
т.ғ.к., доцент, ORCID: 0000-0003-3827-179X,
e-mail: nseilova@iitu.edu.kz
[3]Охрименко Т.А.
т.ғ.к., доцент, ORCID: 0000-0001-9036-6556,
e-mail: t.okhrimenko@nau.edu.ua

[1]Әл-Фараби атындағы Қазақ ұлттық университеті, Алматы қ., Қазақстан
[2]Халықаралық ақпараттық технологиялар университеті, Алматы қ., Қазақстан
[3]Ұлттық авиация университеті, Киев қ., Украина

СЫЗЫҚТЫҚ ҚАБАТТЫ ҚҰРУ ӘДІСІ ТУРАЛЫ

Аңдатпа

Бұл жұмыс блоктық симметриялық шифрлау алгоритмдерінің диффузиялық қабатында қолданылатын сызықтық функциялардың параметрлерін құрудың автоматтандырылған әдісін ұсынады. Негізгі назар тек циклдік ауыстыру операциялары мен биттік XOR-дан жасалған сызықтық қабаттарды жобалауға бағытталған, олар тиімді және аппараттық құралдарға ыңғайлы. Мұндай қабаттар негізгі криптографиялық талап болып табылатын күшті диффузияға қол жеткізуде маңызды рөл атқарады. Ұсынылған әдіс ауысым мәндерін толық санау, олардың тармақтарының санын есептеу және көшкін сипаттамаларын бағалау арқылы үміткер конфигурацияларын бағалайды. Таңдау процесін бағыттау үшін сандық диффузиялық көрсеткіштер жинағы енгізіледі, соның ішінде бір және көп айналымды көшкін әсерлері мен байт деңгейіндегі белсендіру жылдамдығы. Салыстырмалы бағалауға мүмкіндік беретін жиынтық сапа функциясы тұжырымдалған. Әзірленген бағдарламалық құрал төрт 32 биттік сөз ретінде өңделген 128 биттік блоктар үшін оңтайлы ауысу параметрлерін анықтады, бұл тек 12 XOR операциясымен 5 тармақ санына жетеді. Ұсынылған тәсіл классикалық және шектеулі платформалар үшін қолайлы жеңіл және қауіпсіз криптографиялық примитивтердің практикалық синтезіне ықпал етеді.

Тірек сөздер: блоктық симметриялық шифрлау алгоритмі, сызықтық қабат, тармақ нөмірі, көшкін эффектісі.

[1]*Горлов Л.В.,
докторант. ORCID 0000-0002-8208-4716,
e-mail: lev.gorlov@gmail.com
[2]Сейлова Н.А.,
к.т.н., доцент, ORCID: 0000-0003-3827-179X,
e-mail: nseilova@iitu.edu.kz
[3]Охрименко Т.А.,
к.т. н., доцент, ORCID: 0000-0001-9036-6556,
e-mail: t.okhrimenko@nau.edu.ua

[1]Казахский национальный университет им. аль-Фараби, г. Алматы, Казахстан
[2]Международный университет информационных технологий, г. Алматы, Казахстан
[3]Национальный авиационный университет, г. Киев, Украина

О МЕТОДЕ ГЕНЕРАЦИИ ЛИНЕЙНОГО СЛОЯ

Аннотация

В данной статье представлен автоматизированный метод генерации параметров линейных функций, используемых в диффузионном слое блочных симметричных алгоритмов шифрования. Основное внимание уделяется проектированию линейных слоев, построенных исключительно на операциях циклического

сдвига и побитовых операциях «исключающее ИЛИ», которые являются эффективными и аппаратно-ориентированными. Такие слои играют решающую роль в достижении сильной диффузии, фундаментального криптографического требования. Предлагаемый метод оценивает конфигурации-кандидаты путем исчерпывающего перечисления значений сдвига, вычисления количества ветвей и оценки их лавинных характеристик. Вводится набор количественных метрик диффузии для управления процессом выбора, включая одно- и многораундовые лавинные эффекты и показатели активации на уровне байтов. Сформулирована агрегированная функция качества для сравнительной оценки. Разработанный программный инструмент определил оптимальные параметры сдвига для 128-битных блоков, обрабатываемых как четыре 32-битных слова, что позволило достичь количества ветвлений 5 всего за 12 операций XOR. Предложенный подход способствует практическому синтезу легковесных и безопасных криптографических примитивов, подходящих как для классических, так и для ограниченных платформ.

Ключевые слова: алгоритм блочного симметричного шифрования, линейный слой, индекс ветвлений, лавинный эффект.

Article submission date: 19.06.2025