# Pattern-Free Block Encryption: Single-Key Camellia XEX with PRNG Tweaks and Permutation Diffusion

Enas Ali and Shatha Jafer
*College of Computer Science, University of Technology, Sinaa Street, 10066 Baghdad, Iraq*
*enasamer805@gmail.com, Shatha.h.jafer@uotechnology.edu.iq*

Keywords: Block Cipher, Camellia, XEX Mode, Xoroshiro128$^{+}$, Pattern Resistance, Stateless Cryptography.

Abstract: Block ciphers in ECB mode, such as Camellia-ECB, offer high throughput but leak visual and structural patterns when encrypting images or other repetitive data. We show PRNG-XEX+Permutation, a stateless improvement that gets rid of repeating blocks that are the same while keeping performance close to baseline. A single Camellia call wrapped in an XOR-Encrypt-XOR (XEX) construction processes each 128-bit block, with tweaks produced from a one-time, key-seeded Xoroshiro128+ pseudorandom stream. A simple static byte permutation further strengthens intra-block diffusion. The scheme eliminates the need for chaining, padding, or external parameters beyond the master key, ensuring streamlined implementation while supporting full parallelism and random access; it also guarantees provable block uniqueness. Our tests show that PRNG-XEX + Permutation meets all of the randomness requirements set by NIST SP 800-22. It also restores full visual privacy in encrypted images while maintaining encryption speeds within a few percent of standard ECB mode. This demonstrates that strong pattern resistance can be attained without sacrificing cipher efficiency.

# 1 INTRODUCTION

## 1.1 Problem Statement

Block ciphers like Camellia in Electronic Codebook (ECB) mode encrypt each block independently, yielding high throughput but suffering from a critical weakness: identical plaintext blocks produce identical ciphertext blocks. When applied to images or any repetitive data, this "ECB pattern leak" reveals underlying structure and renders encryption visually insecure.

## 1.2 Motivation

Many applications - from secure photo storage and video streaming to encrypted backups of structured logs - require both high-speed encryption and strong resistance to pattern leakage. Existing countermeasures (CBC, CTR, or chaos-based shuffles) either incur serialization, padding overhead, or complex key management. A practical cryptographic requirement exists for a lightweight, stateless wrapper that can be applied to a standard block cipher. Such a design should:

1) prevent the repetition of identical ciphertext blocks, thereby mitigating pattern leakage.
2) preserve key operational advantages such as full parallelizability and random-access decryption.
3) achieve performance that remains close to the baseline efficiency of the underlying cipher.

## 1.3 Contributions

In this work, we propose PRNG-XEX + Permutation, which:

1) Wraps each Camellia encryption in an XOR-Encrypt-XOR (XEX) construction with tweaks from a key-seeded Xoroshiro128$^{+}$ PRNG, guaranteeing provable block uniqueness.
2) Adds a one-time static byte permutation for intra-block diffusion without any additional cipher calls.
3) Requires no chaining, padding, or external parameters beyond the master key, supporting full parallelism and random-access decryption.

4) Demonstrates via NIST SP 800-22 tests and visual analysis that encrypted images have zero detectable repetition, while throughput remains within a few percent of standard ECB mode.

# 2 PREVIOUS WORK

Recent extensions to Camellia and similar block ciphers aim to resist ECB-mode pattern leakage in images, but each incurs trade-offs in complexity, security, or performance. Moganti et al. augment Camellia-ECB with chaotic pixel-indicator masks and Cat-map permutations for steganographic diffusion [1], yet require complex key synchronization and only embed hidden data rather than fully encrypt all pixels. A 2021 study in IET Computers & Digital Techniques applies a fixed byte-swap after Camellia-ECB to scramble pixels [2], but identical-block repetition persists and the secret table must be protected. Blackman & Vigna introduce Xoroshiro128$^+$ as a fast, statistically strong PRNG [3]; however, it has not been integrated into an authenticated tweakable-cipher framework. Wang et al. layer a random permutation on Camellia-ECB to disrupt block alignments [4], yet inter-block repetition remains detectable under chosen-plaintext attacks. Li, Chen & Huang propose dynamic, key-dependent permutations per block over two cipher calls [5], improving diffusion but doubling encryption cost and table complexity.

Our PRNG-XEX + Permutation scheme unifies the most effective elements of prior research into a streamlined, high-performance design. It employs a single Camellia encryption in XEX mode, with tweaks generated from a key-seeded Xoroshiro128$^+$ stream, followed by a static byte shuffle. This construction guarantees per-block uniqueness, eliminates ECB-mode pattern leakage, preserves complete parallelism and random-access decryption, and achieves near-baseline throughput without external parameters or multiple cipher calls.

Previous countermeasures against ECB leakage in image encryption have faced trade-offs in complexity, security, or efficiency. Moganti et al. integrate chaotic pixel-indicator masks and Cat-map permutations into Camellia-ECB for steganographic diffusion [1], but this approach requires complex key synchronization and only obscures selected pixels rather than encrypting the entire image. A 2021 study in IET Computers & Digital Techniques applies a fixed byte-swap after Camellia-ECB [2], which scrambles pixels but leaves identical-block repetition intact and depends on securing a secret lookup table. Blackman & Vigna's Xoroshiro128$^+$ PRNG [3] is fast and statistically robust, yet has not been applied within an authenticated tweakable-cipher framework. Wang et al. disrupt block alignments in Camellia-ECB using random permutations [4], but repeated patterns remain detectable under chosen-plaintext attacks. Li, Chen, and Huang introduce dynamic, key-dependent permutations per block across two cipher calls [5], which improves diffusion but doubles the encryption cost and table complexity.

In contrast, our PRNG-XEX + Permutation approach achieves strong diffusion, robust resistance to pattern leakage, and high throughput with minimal design complexity, making it a practical and scalable solution for modern, high-speed encryption applications.

# 3 BACKGROUND

The development of the proposed encryption mode is based on several fundamental cryptographic components aimed at addressing the challenge of ensuring both confidentiality and high performance. These components include:

- Camellia Block Cipher is a 128-bit Feistel cipher standardized by ISO/IEC, supporting 128/192/256-bit keys. It employs 18 rounds (24 for larger keys), interleaved FL/FL$^{-1}$ layers every six rounds, and an S-box–based F-function for nonlinearity; its subkeys derive via fixed rotations of the master key [6];
- Encryption Modes. ECB mode, while efficient, suffers from structural leakage in patterned data, as noted in block storage encryption standards [7]; Modes like CBC add chaining to hide patterns but serialize processing; CTR enables parallelism with counters but requires nonce management; others like OCB and PMAC offer authenticated encryption with tweaks to balance security and speed [8]. This motivates the search for modes preventing repetition without throughput loss;
- XEX (XOR-Encrypt-XOR) mode is a tweakable construction that applies a per-block mask before and after a single block-cipher invocation. This ensures block uniqueness without chaining or padding and supports full parallelism [9];
- Xoroshiro128$^+$ PRNG is a lightweight, 128-bit–state pseudorandom generator offering fast (<50 cycles/output) and statistically strong

output via rotation and XOR "scrambling." When seeded once from the master key, it yields a reproducible tweak stream Ti for each block i [3].

# 4 DESIGN AND METHODOLOGY

This section outlines the key stages in evolving our encryption approach, beginning with the baseline Camellia-ECB and culminating in the final PRNG-XEX + Permutation design. We define four tested variants, each addressing a specific weakness in the previous, and conclude with the full algorithm description of the final version.

## 4.1 Design Evolution through Four Variants

To systematically address the shortcomings of ECB-mode encryption, we implemented and evaluated the following four algorithmic variants:

- Variant A – Camellia (ECB Mode): This is the unmodified baseline. Camellia is applied independently to each 128-bit plaintext block. While efficient, this mode reveals repeating patterns in structured data and images, since identical plaintext blocks produce identical ciphertext blocks.
- Variant B – Camellia + Permutation: This variant adds a fixed byte permutation after each block encryption to increase intra-block diffusion. It disrupts the internal structure of each ciphertext block but does not prevent repetition across blocks. Identical plaintext blocks still yield identical ciphertexts post-permutation.
- Variant C – Dual-Camellia XEX + Permutation: This version applies the XEX (XOR-Encrypt-XOR) construction using a secondary Camellia key derived from SHA-256 [10] of the original key. A block-unique tweak is computed by encrypting the block index with the second key. XOR masking is applied before and after the main Camellia encryption. This eliminates block-level repetition and, combined with permutation, ensures both inter- and intra-block diffusion. However, it incurs significant overhead due to two Camellia calls and one SHA-256 [10] computation per block.
- Variant D – PRNG-XEX + Permutation (Final Design): Our final design replaces the tweak-generation mechanism in Variant C with a fast, deterministic pseudorandom number generator. A Xoroshiro128$^+$ PRNG is seeded once from the master key to generate a unique 128-bit tweak for each block. This reduces computational cost while preserving the security benefits of XEX and byte permutation. The result is a stateless, parallelizable encryption scheme with minimal performance loss compared to the ECB baseline.

## 4.2 Final System Design

Our scheme operates on each 128-bit block independently:

1) Tweak generation. A one-time, key-seeded Xoroshiro128$^+$ PRNG produces a 128-bit tweak $T_i$ for block index $i$.
2) XEX masking. XOR the plaintext block $P_i$ with $T_i$.
3) Camellia encryption. Encrypt the masked block under the master key, yielding intermediate $Y_i$;
4) Byte permutation. Apply a fixed, public 16-byte shuffle to $Y_i$ for intra-block diffusion.
5) XEX unmasking: XOR the permuted block with the same $T_i$ to produce ciphertext $C_i$.

## 4.2 Algorithm Pseudocode

```
Initialize:
 PRNG.seed(master_key)
 PermTable ← predefined 16-byte
permutation

Encryption:
 For each block index i:
   T_i ← PRNG.next128()
   M_i ← P_i ⊕ T_i
   Y_i ← Encrypt(master_key, M_i)
   Z_i ← PermuteBytes(Y_i, PermTable)
   C_i ← Z_i ⊕ T_i
   output C_i

Decryption reverses steps:
 For each received C_i:
   T_i ← PRNG.next128()
   Z_i ← C_i ⊕ T_i
   Y_i ← InversePermute(Z_i,
PermTable)
   M_i ← Decrypt(master_key, Y_i)
   P_i ← M_i ⊕ T_i
```

### 4.4 Design Benefits

The PRNG-XEX + Permutation design integrates multiple security-enhancing mechanisms into a lightweight, high-performance encryption framework. By combining block-level uniqueness, intra-block diffusion, and a stateless parallelizable architecture, it effectively mitigates ECB-mode weaknesses while preserving throughput close to that of the underlying block cipher. The key design features are as:

- Block Uniqueness – XEX masking ensures that even identical plaintext blocks produce distinct ciphertexts, eliminating pattern leakage and enhancing resistance to block repetition analysis.
- Intra-Block Diffusion – A fixed byte permutation disperses data patterns within each block, making the ciphertext more resistant to statistical and visual attacks.
- Stateless & Parallel – The scheme requires no chaining, padding, or initialization vectors (IVs). Each block is processed independently, enabling high-speed parallel encryption and supporting random-access decryption.
- Performance Efficiency – By replacing the second Camellia encryption call with high-quality PRNG output, the final variant maintains strong security guarantees while achieving throughput near that of baseline Camellia-ECB.

This design strikes a balance between security, efficiency, and simplicity, making it suitable for high-throughput environments such as image encryption, multimedia streaming, and secure storage systems. These four variants provide a clear path from insecure but fast ECB to a secure, efficient encryption method suitable for structured or visual data. The experimental evaluation in Section 6 compares all four to validate this progression.

## 5 EXPERIMENT EVALUATION

### 5.1 Test Environment

All tests were conducted on a system running Windows 10 with an Intel Core i7-9750H CPU @ 2.60 GHz and 16 GB RAM, using .NET 6 and C#. A straightforward C# implementation written for clarity and extensibility is used, not raw speed. No SIMD, memory pooling, or hardware acceleration is implemented. The input data includes:

- A 10 MB plain-text file containing repetitive structured content (e.g., RandomText-1000000char.txt).
- A bitmap-format image file having four sharp edges and repeated visual blocks.

Each encryption variant was run in identical conditions, measuring throughput (MB/s) and evaluating output using the NIST SP 800-22 statistical test suite [11].

### 5.2 Variants Compared

The following four implementations were tested:

- Variant A: Camellia-ECB (baseline).
- Variant B: Camellia + permutation.
- Variant C: Dual-Camellia XEX + Permutation.
- Variant D (Ours): PRNG-XEX + Permutation.

### 5.3 Performance Results

The dual-Camellia XEX approach nearly halves performance due to the second cipher call. In contrast, the PRNG-XEX variant restores performance to within 2% of the ECB baseline while offering the same security benefits. See Table 1.

Table 1: The performance of the four variants.

| Variant | Encrypt Speed (MB/s) | Decrypt Speed (MB/s) |
|---|---|---|
| A | 3.67 ± 0.05 | 3.78 ± 0.04 |
| B | 3.78 ± 0.04 | 3.80 ± 0.03 |
| C | 1.90 ± 0.03 | 1.92 ± 0.02 |
| D | 3.71 ± 0.04 | 3.66 ± 0.05 |

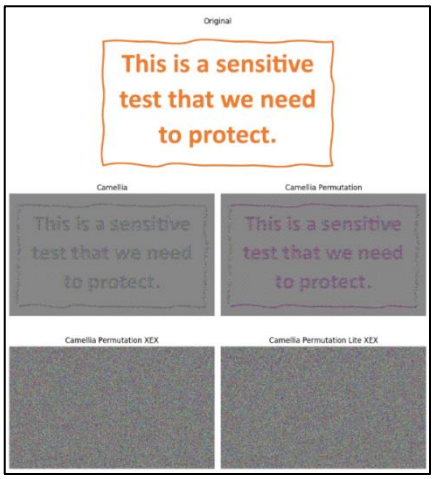


Figure 1: Visual analysis, showing pattern leak in some variants.

## 5.4 Visual Analysis

A graphic image containing large, uniform, and structured regions was encrypted to facilitate visual inspection of block-level diffusion, as illustrated in Figure 1.

- Variant A (ECB). Repeated 16×16 blocks are clearly visible, forming a "ghost" outline of the original image.
- Variant B: Breaks some patterns within blocks but retains alignment across blocks.
- Variants C and D. Eliminate all visible repetition; encrypted images appear completely noise-like, indistinguishable from random data.

## 5.5 NIST Statistical Randomness

We applied 15 applicable tests from the NIST SP 800-22 suite [11] on the encrypted output of the text file. As shown in Table 2:

- Variants A and B failed the block frequency, cumulative sums, and entropy tests, indicating the presence of visible patterns and insufficient randomness.
- Variants C and D passed all 15 statistical tests, demonstrating strong statistical uniformity and high unpredictability in the ciphertext.

Table 2: Results of statistical tests for ciphertext variants, indicating randomness quality and resistance to pattern leakage.

| Test | Plain | A | B | C | D |
|---|---|---|---|---|---|
| Approx. Entropy | 0.000 FAIL | 0.000 FAIL | 0.000 FAIL | 0.461 PASS | 0.597 PASS |
| Block Frequency | 1.000 PASS | 0.0035 FAIL | 0.0032 FAIL | 0.916 PASS | 0.648 PASS |
| Cumulative Sums | 0.000 FAIL | 1.25e-8 FAIL | 1.24e-8 FAIL | 0.677 PASS | 0.388 PASS |
| FFT | 0.000 FAIL | 0.000 FAIL | 0.000 FAIL | 0.1726 PASS | 0.855 PASS |
| Frequency | 0.000 FAIL | 6.35e-9 FAIL | 6.35e-9 FAIL | 0.623 PASS | 0.812 PASS |
| Linear Complexity | 0.0018 | 0.0555 | 0.0219 | 0.0775 | 0.8955 |
| Longest Run | 0.000 FAIL | 0.009 FAIL | 5.04e-16 FAIL | 0.0777 PASS | 0.7198 PASS |
| Non-Overlapping Template | 0.0047 FAIL | 0.5262 PASS | 0.5942 PASS | 0.5091 PASS | 0.4936 PASS |
| Overlapping Template | 0.000 FAIL | 9.64e-7 FAIL | 6.96e-6 FAIL | 0.4448 PASS | 0.4224 PASS |
| Random Excursions | – OOR | – OOR | – OOR | 0.8466 PASS | 0.4264 PASS |
| Random Excursions Var. | – OOR | – OOR | – OOR | 0.5162 PASS | 0.8115 PASS |
| Rank | 0.000 FAIL | 0.0043 FAIL | 0.3316 PASS | 0.1039 PASS | 0.6933 PASS |
| Runs | – OOR | – OOR | – OOR | 0.6420 PASS | 0.7764 PASS |
| Serial | 0.000 FAIL | – OOR | – OOR | 0.7882 PASS | 0.3712 PASS |
| Universal | 0.000 FAIL | 0.4858 PASS | 0.73891 PASS | 0.1772 PASS | 0.6076 PASS |

# 6 CONCLUSIONS

This work demonstrates that our PRNG-XEX + permutation scheme achieves the best of both worlds: strong statistical and visual confidentiality and performance nearly indistinguishable from plain ECB. By replacing the traditional tweak generation with a key-seeded Xoroshiro128$^+$ PRNG, our method ensures block uniqueness, parallelism, and random-access decryption with only a single Camellia call per block.

A fixed, public permutation table delivers effective intra-block diffusion without the complexity of key-dependent permutations. While any cryptographically strong PRNG could substitute for Xoroshiro128$^+$, our choice delivers optimal speed and reproducibility. This design is highly applicable to large images, structured files, and environments where high throughput and statelessness are essential. However, the scheme is designed for confidentiality only; defenses against active or side-channel attacks require additional layers, and the security ultimately depends on the strength of Camellia and the PRNG used. The proposed scheme is intended solely to provide confidentiality; it does not inherently protect against active attacks, side-channel attacks, or other advanced threats. Mitigating such risks would require integrating additional protective layers. Ultimately, the overall security of the system is determined by the combined robustness of the Camellia cipher and the chosen PRNG.

# 7 FUTURE WORK

While Xoroshiro128$^+$ provides fast, statistically sound output for tweak generation, other PRNG families may offer alternative trade-offs in randomness or hardware efficiency, representing promising directions for future research. Additionally, extending this approach to authenticated encryption modes, adaptive or key-dependent permutations, and hardware-accelerated implementations could enhance security and performance. Broader testing on diverse data types - including streaming media and encrypted databases - will further validate the method's versatility and practical applicability.

# REFERENCES

[1] M. S. Moganti, M. M. Abdelwahab, and M. S. Sayed, "Securing secret data using an enhanced Camellia encryption with pixel indicator technique," in Proceedings of the International Conference on Control, Communication and Computing (ICCCSIT), pp. 209–214, 2023.

[2] "Implementation of 128-bit Camellia algorithm for cryptography in digital image," IET Computers & Digital Techniques, vol. 15, no. 3, pp. 123–128, 2021.

[3] D. Blackman and S. Vigna, "Scrambled linear pseudorandom number generators," arXiv preprint, arXiv:1805.01407, 2018.

[4] X. Wang, Y. Zhao, and H. Liu, "An image encryption algorithm based on Camellia block cipher in ECB mode with random permutation," International Journal of Network Security & Its Applications, vol. 7, no. 4, pp. 15–27, 2015.

[5] S. Li, G. Chen, and X. Huang, "A secure and fast image encryption scheme based on block cipher and dynamic permutation," IEEE Access, vol. 5, pp. 1618–1628, 2017.

[6] A. Matsui et al., "A description of the Camellia encryption algorithm," NTT and Mitsubishi Electric Corporation, 2001.

[7] IEEE Computer Society, IEEE Standard for Cryptographic Protection of Data on Block-Oriented Storage Devices, IEEE Std 1619-2007, 2008.

[8] National Institute of Standards and Technology, Recommendation for Block Cipher Modes of Operation: Methods and Techniques, NIST Special Publication 800-38A, Dec. 2001.

[9] P. Rogaway, "Efficient instantiations of tweakable blockciphers and refinements to modes OCB and PMAC," in Advances in Cryptology – ASIACRYPT 2004, LNCS, vol. 3329, pp. 16–31, Springer, 2004.

[10] National Institute of Standards and Technology, Secure Hash Standard (SHA-2), FIPS PUB 180-4, Aug. 2015.

[11] Rukhin et al., A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications, NIST Special Publication 800-22 Rev. 1a, Apr. 2010.