# STORY2: A Fully Parameterised, Story-Key Driven Substitution-Permutation Network Cipher

Design, Novel Diffusion Construction, and Empirical Analysis

Nabil Islam

Independent Researcher

*w3nabil@gmail.com*

**Abstract**

We present STORY2, a symmetric block cipher built on a substitution-permutation network (SPN) architecture in which every cryptographic parameter — including the S-box selection and all round keys — is derived deterministically from a natural-language narrative supplied by the user as the primary key. This paper introduces a novel $16 \times 16$ Cauchy Maximum Distance Separable (MDS) diffusion matrix over $\mathbb{GF}_{2^8}$. The matrix achieves a branch number of 17, the theoretical maximum for any $16 \times 16$ linear map over $\mathbb{GF}_{2^8}$, and enforces complete 16-byte diffusion in a single pass. The construction is proven invertible by the Cauchy determinant theorem. To our knowledge, no published block cipher employs a full-state $16 \times 16$ Cauchy MDS matrix as its sole diffusion layer. We also introduce a two-level, privately held S-box pool architecture that functions as an independent secondary security factor: if the story key is disclosed, a private pool whose contents are unknown to the adversary requires a brute-force search of $256! \approx 2^{1684}$ bijections for decryption. Empirical analysis across 2,000 story keys confirms near-ideal statistical behaviour: avalanche effect mean of 50.007%, and SAC convergence at round 1 — four rounds earlier than an equivalent $4 \times 4$ column-restricted baseline.

**Keywords:** block cipher, substitution-permutation network, narrative key, story-based cryptography, Cauchy MDS matrix, $\mathbb{GF}_{2^8}$ arithmetic, full-state diffusion, branch number, key-dependent S-box, SHAKE-256, HMAC, CTR mode.

# Contents

# 1. Introduction

## 1.1. Problem Statement

Modern symmetric encryption systems are cryptographically strong but place a significant burden on the user to manage key material. A randomly generated key — for example, *3a7f9c1d 8e2b4f05 6d0a3c7e 9b1f5d2a* — is not memorable and must be stored in a password manager, a file, or a hardware token. If that storage is lost or compromised, the encrypted data cannot be recovered. Password-based systems exist, but they treat the password as a low-quality secret that must be expanded into a proper key using computationally expensive functions such as PBKDF2 [13] or Argon2 [5]. These functions address low entropy by making each guess expensive, but they do not increase the unpredictability of the password itself, and users persistently choose weak, predictable secrets [6].

## 1.2. Story2 Approach

Story2 treats the key differently. Rather than asking for a random string or a short password, it accepts a personal *narrative* — any sentence, paragraph, or passage that the user can write naturally in their own language and recall from memory. An example is: *"The deepest darkest secret of my life is I had a crush on my junior school's maths teacher. Her name is Zuhan Shu and she is currently advancing her PhD in medical imaging technology."* This narrative is easy for the author to reconstruct but difficult for an adversary to guess, because the attacker must recover not only the words but also the private facts, the personal context, and the precise phrasing — including any typographical or stylistic idiosyncrasies — that the author chose [7]. Crucially, Story2 is not security through obscurity. The algorithm is fully public; security derives from the mathematical properties of the cipher and the unpredictability of the narrative, not from hiding how the design works. The narrative is encoded in UTF-16-LE and processed through a two-step HMAC-SHA256 key derivation pipeline to produce a 256-bit encryption key and a 256-bit authentication key (Section 8.2).

## 1.3. Origins

The idea for Story2 emerged from a fictional spy novel written by the author in 2022, in which characters communicated through coded narratives. The fictional system used simple character-index rotation and was clearly insecure. In 2023, during separate research into mail-based attack vectors, the need for a memorable, high-entropy key primitive led to a systematic redesign: the narrative concept was retained, whilst the underlying cryptographic structure was replaced with a proper SPN incorporating S-boxes, an MDS diffusion matrix, round keys, and authenticated encryption. After three years of development, Story2 is now a formally analysed, empirically tested cipher.

### 1.4. Research Questions

This paper addresses the following six research questions.

1. Under what conditions does a human-authored narrative provide guessing entropy sufficient for cryptographic key material, and how do encoding choice, writing style, and adversarial context knowledge affect that entropy?

2. Does a $16 \times 16$ Cauchy MDS matrix provide strictly better diffusion than the column-restricted $4 \times 4$ design, and what is the measurable difference in SAC convergence and computational cost?

3. Is there a condition on MDS matrix dimension $n$ such that a full-state $n \times n$ MDS layer makes a separate permutation layer cryptographically redundant in a standard SPN, and does the $16 \times 16$ Cauchy construction satisfy it?

4. Does the Encrypt-then-MAC construction [3] correctly provide IND-CCA2 confidentiality and INT-CTXT authenticity, and does it detect tampering in practice across all tested modification types?

5. Under what conditions does a privately held S-box pool constitute a genuine independent security layer, and what are the precise boundaries of that guarantee with respect to adversary capability?

6. Is a full-state $16 \times 16$ Cauchy MDS cipher practically viable at software speeds, and how does its throughput compare with an equivalent $4 \times 4$ column-restricted baseline at the same security level?

Questions 1, 2, and 6 are addressed through empirical analysis in Section 11. Questions 3 and 5 are addressed through formal argument in Sections 7 and 5 respectively. Question 4 is addressed through both formal design argument and a targeted tamper-detection test suite described in Section 8.8.

### 1.5. Contributions

This paper makes the following contributions.

- A complete SPN cipher in which every cryptographic parameter — the S-box selection, the round keys, and the whitening key — is derived deterministically from a natural-language narrative, validated empirically across 2,000 story keys in multiple languages and scripts.

- An application of the Cauchy determinant theorem to the $16 \times 16$ case, proving that every matrix constructed by the described method is invertible and MDS without numerical verification.

- A formal proof that a full-state 16×16 MDS matrix with branch number 17 makes a separate permutation layer cryptographically redundant in an SPN, reducing the round function to three operations without weakening any diffusion property. To our knowledge, no published cipher has stated this as a general design principle.
- A measured demonstration that the $16 \times 16$ MDS achieves full Strict Avalanche Criterion convergence at round 1, confirmed across 2,000 story keys.
- **A two-factor security model via a private S-box pool.** A formal proof that an independently held S-box pool, whose contents are orthogonal to the story key, constitutes a genuine second security factor: passive recovery of the story key does not yield decryption, and the adversarial search cost is $256! \approx 2^{1684}$. The precise boundary at which this guarantee fails — oracle access to the encryption function — is derived explicitly.
- A formal argument that encoding the narrative as UTF-16-LE provides a uniform $2^{16} = 65{,}536$ symbol space per character position regardless of script, compared with 128 for ASCII characters in UTF-8, eliminating encoding-based entropy penalties for non-Latin writers.
- Exact computation of $\mathrm{DDT}_{\max} = 4$, $\mathrm{NL} = 112$, $\mathrm{BCT}_{\max} = 6$, $\mathrm{BN} = 17$, differential security of 390 bits, linear security of 262 bits, boomerang security of 600 bits, and nine further results covering algebraic degree, integral resistance, impossible differentials, slide resistance, key schedule security, and AEAD security — all computed by Python script and independently confirmed by MILP.
- An Encrypt-then-MAC construction using CTR mode and HMAC-SHA256 with independently derived encryption and authentication keys, providing both IND-CCA2 confidentiality and INT-CTXT authenticity with a 256-bit authentication tag.

### 1.6. Paper Structure

Section 2 surveys related work. Section 3 presents the mathematical background. Section 7 develops the Cauchy MDS matrix in detail. Section 4 analyses the narrative key model. Section 5 presents the private S-box pool as a secondary security layer. Section 8 describes the full cipher design. Section 6 gives the formal security analysis. Section 9 discusses limitations and future work. Section 11 concludes. The empirical test results are presented in the Appendix.

## 2. Related Work

### 2.1. Password Security and Passphrase-Based Key Material

The security of an encryption system depends critically on the quality of the secret used to derive keys. Bonneau [6] analysed nearly 70 million user passwords and found that real-world passwords

provide fewer than ten bits of security against online attacks. This gap between theoretical key strength and human behaviour is well-documented: users prefer short, predictable strings that are easy to type but straightforward to guess.

Passphrases — sequences of natural-language words — have been proposed as a more memorable alternative. The Diceware scheme [25] formalises this by selecting words uniformly at random from a known word list, producing keys whose entropy is calculable. The Bitcoin BIP-39 standard [24] applies the same principle to wallet recovery phrases. Both approaches demonstrate that word-level key material is practical at scale.

However, Bonneau and Shutova [7] show that user-generated multi-word passphrases fall well short of randomly sampled word lists in entropy: natural language has strong statistical structure, and people choose predictable phrases. Shay et al. [30] further found that system-assigned random passphrases offer no measurable usability advantage over random character passwords.

STORY2 departs from both approaches. It uses a personal narrative — a story the user already knows — rather than a sampled or assigned phrase. The entropy of such a narrative cannot be estimated from a word-list frequency table alone, because the attacker must also reconstruct the private facts, the specific writing style, and the personal context behind it. The key derivation pipeline maps this variable-length narrative to a fixed-length key regardless of script or language (Section 8.2).

### 2.2. Block Cipher Design: Confusion, Diffusion, and the SPN

Modern block ciphers are built on Shannon's principles of confusion and diffusion [28]. Confusion is provided by non-linear substitution layers (S-boxes); diffusion is provided by linear mixing layers that spread the influence of each input bit across the full cipher state. The substitution-permutation network (SPN) organises these operations into iterated rounds.

The Strict Avalanche Criterion (SAC), introduced by Webster and Tavares [32], defines diffusion quality: a function satisfies SAC if flipping any single input bit causes each output bit to change with probability exactly one-half. SAC convergence round is used in this paper as the primary metric for comparing diffusion layers.

STORY2 replaces the column-restricted $4 \times 4$ design with a $16 \times 16$ Cauchy MDS matrix operating over the entire 128-bit state simultaneously. The theoretical basis for Cauchy MDS matrices is well established [11, 20]: every square submatrix of a Cauchy matrix over a suitable finite field is non-singular, so the MDS property is guaranteed by construction. Recent work [17] shows that matrices of dimension 16 can be implemented efficiently in software using lookup-table techniques. The empirical comparison between the $16 \times 16$ and $4 \times 4$ designs is presented in Section 8.

### 2.3. Key Derivation and Extendable-Output Functions

Key derivation functions such as PBKDF2 [13] and Argon2 compensate for low-entropy passwords by making each guess expensive. They do not increase the entropy of the password itself.

STORY2 does not use a password-stretching KDF. It relies on the narrative providing sufficient guessing entropy directly, and uses SHAKE-256 [21] as a variable-input, fixed-output extendable function to produce key material of any required length. SHAKE-256, standardised in NIST FIPS 202, provides 256-bit security against collision and preimage attacks.

### 2.4. Authenticated Encryption and the Encrypt-then-MAC Construction

Encryption alone provides confidentiality but not integrity. Authenticated Encryption addresses this by binding a MAC to the ciphertext so that any modification is detected before decryption.

Bellare and Namprempre [3] proved that only the Encrypt-then-MAC composition reliably achieves both IND-CCA2 privacy and integrity of ciphertexts. STORY2 uses Encrypt-then-MAC with CTR-mode encryption followed by HMAC-SHA256 over the nonce and ciphertext. The tag is verified before any decryption is attempted, ensuring that no partial plaintext is ever produced from a tampered message.

### 2.5. Full-State SPN Diffusion: Related Designs

Several published ciphers use full-state MDS diffusion. SHARK [8] uses an $8 \times 8$ MDS matrix over GF($2^8$) without a separate permutation layer. WHIRLPOOL [1] uses an $8 \times 8$ MDS-based mixing matrix in a Miyaguchi-Preneel construction. PHOTON [10] uses $4 \times 4$ and $8 \times 8$ MDS matrices in lightweight settings. None of these designs employ a $16 \times 16$ Cauchy MDS matrix as the sole diffusion layer of a 128-bit block cipher. STORY2's construction is therefore novel in this specific regard.

## 3. Mathematical Background

### 3.1. Galois Field Arithmetic

The diffusion layer of STORY2 operates over $\mathbb{GF}_{2^8}$, the field of $2^8 = 256$ elements. Elements are represented as degree-at-most-7 polynomials with coefficients in $\{0, 1\}$, equivalently as single bytes. Addition in $\mathbb{GF}_{2^8}$ is bitwise XOR. Multiplication is polynomial multiplication modulo the irreducible polynomial

$$p(x) = x^8 + x^4 + x^3 + x + 1, \tag{1}$$

which is the same polynomial used in standard SPN constructions [9]. Every non-zero element $a$ has a unique multiplicative inverse $a^{-1}$ satisfying $a \cdot a^{-1} = 1$.

In the reference implementation a full $256 \times 256$ multiplication table `_GF` is precomputed at import time using the shift-and-XOR algorithm, so that `_GF[a][b]` returns $a \cdot b$ in $O(1)$ time

with no runtime polynomial arithmetic.

### 3.2. Sponge Functions and SHAKE-256

SHAKE-256 [22] is an extendable-output function from the SHA-3 family, built on the Keccak permutation over a 1,600-bit state. Its 512-bit capacity gives SHAKE-256 a 256-bit security level against collision and preimage attacks.

STORY2 uses SHAKE-256 to compress the story string to a fixed-length pseudorandom byte stream regardless of narrative length, and to produce the byte streams required for S-box index derivation.

### 3.3. HMAC and Domain-Separated Key Derivation

HMAC [15] is defined as

$$\mathrm{HMAC}(K,\, m) \;=\; H\big((K \oplus \mathit{opad}) \,\|\, H((K \oplus \mathit{ipad}) \,\|\, m)\big). \tag{2}$$

Under the assumption that $H$ is a PRF, HMAC is also a PRF.

STORY2 uses HMAC-SHA256 in an HKDF-expand-style pattern [16]. A single 32-byte pseudorandom key (`prk`) is first derived from the story string; all subsequent subkeys are then obtained via domain-separated HMAC calls using distinct ASCII labels. Table 1 lists every label used in the implementation.

**Table 1.** HMAC domain-separation labels used in STORY2.

| Label | Derived value |
|---|---|
| `story_v2_salt` | Pseudorandom key `prk` |
| `enc\|\|story_v2_master\x01` | Encryption subkey $K_{\mathrm{enc}}$ |
| `mac\|\|story_v2_master\x02` | Authentication subkey $K_{\mathrm{mac}}$ |
| `story_v2_keys\|\|` | Round keys (80 bytes, 5×16) |
| `story_v2_whitening\|\|` | Final whitening key (16 bytes) |
| `story_v2_sbox\|\|` | S-box selection index |

This scheme ensures $K_{\mathrm{enc}} \perp K_{\mathrm{mac}}$, and more broadly that no two derived values share a label, so compromise of one output gives no information about any other.

### 3.4. Maximum Distance Separable Matrices

The *branch number* of a linear map $M$ is defined as

$$\mathcal{B}(M) \;=\; \min_{\mathbf{s}\neq\mathbf{0}}\big(\mathrm{wt}(\mathbf{s}) + \mathrm{wt}(M\mathbf{s})\big),$$

where wt($\mathbf{v}$) counts the number of non-zero byte positions in $\mathbf{v}$. A higher branch number implies stronger diffusion: a single non-zero input byte forces more non-zero output bytes, which reduces the probability of useful differential characteristics.

The theoretical maximum branch number for an $n \times n$ byte-oriented matrix is $n + 1$. A matrix achieving this maximum is called a *Maximum Distance Separable* (MDS) matrix. For STORY2's $16 \times 16$ matrix the maximum is $16 + 1 = 17$, meaning any single non-zero input byte propagates to all 16 output bytes in a single application.

## 4. The Narrative Key Model

This section addresses three research questions raised in Section 1.4: whether a personal narrative provides sufficient guessing entropy to serve as primary key material (RQ1), how the UTF-16-LE encoding choice affects the per-character guessing cost compared with UTF-8 (RQ1), and what the security guarantee is when an adversary has partial knowledge of the narrative's context (RQ1).

### 4.1. Threat Model

We consider two distinct adversarial settings, as the security claim differs substantially between them.

**Stranger adversary.** The adversary has no prior knowledge of the story author and must reconstruct the narrative from the ciphertext alone, guided only by the statistical properties of natural language. Against this adversary the effective key space is bounded below by Shannon's entropy estimate for written English: approximately 1.0–1.3 bits per character [29]. A 30-word narrative contains roughly 150 characters (mean English word length $\approx 4.7$ characters plus one space [23]), giving a conservative guessing entropy of

$$H_{\text{stranger}} \;\geq\; 1.0 \times 150 \;=\; 150 \text{ bits},$$

which already exceeds the 128-bit security target of a standard symmetric cipher. In practice, personal narratives that include private facts (a specific person's name, a secret, a private memory) have considerably higher entropy than general English text, because the adversary must reconstruct both the linguistic form and the underlying private facts simultaneously.

**Insider adversary.** The adversary has partial contextual knowledge about the author: for example, knowing that the author is a chess player, or that the narrative concerns a particular life event. This knowledge constrains the effective search space. STORY2 does not claim resistance to an adversary with complete contextual knowledge; security in this setting reduces to the unpredictability of the specific phrasing, private details, and personal writing style within the

known context. We therefore recommend, in Section 4.4, that narrative keys include at least one piece of information that is not known to any potential adversary, regardless of their relationship to the author.

### 4.2. Encoding Choice: UTF-16-LE versus UTF-8

STORY2 encodes the narrative as UTF-16-LE bytes before any cryptographic operation. This choice is deliberate and has a measurable effect on the per-character guessing cost.

In UTF-8, every ASCII character (a–z, 0–9, common punctuation) encodes to exactly one byte, with 128 possible values in the range `0x00`–`0x7F`. Non-ASCII characters require 2–4 bytes, but the valid byte sequences are constrained by the UTF-8 encoding rules, so the effective symbol space per character is not the full $2^{16}$ or $2^{32}$.

In UTF-16-LE, every character in the Basic Multilingual Plane (BMP) — which covers all characters a person would write in any everyday language, including Latin, Arabic, Chinese, Japanese, Korean, and Cyrillic scripts — encodes to exactly two bytes, giving $2^{16} = 65{,}536$ possible values per character position. Even a simple lowercase ASCII letter such as `a` (U+0061) becomes the two-byte sequence `61 00`, contributing both bytes to the SHAKE-256 input.

Table 2 summarises the per-character symbol space under each encoding.

**Table 2.** Per-character symbol space under UTF-8 and UTF-16-LE.

| Encoding | Bytes/char | Symbol space | $\log_2$(space) |
|---|---|---|---|
| UTF-8 (ASCII range) | 1 | 128 | 7 bits |
| UTF-8 (non-ASCII BMP) | 2–3 | $< 65{,}536$ | <16 bits (constrained) |
| UTF-16-LE (any BMP) | 2 | 65,536 | 16 bits |

UTF-16-LE provides a uniform 16 bits of symbol space per character regardless of script, compared with 7 bits for ASCII characters in UTF-8. This means the raw symbol-space upper bound for a $c$-character narrative is $65{,}536^c = 2^{16c}$ under UTF-16-LE, versus $128^c = 2^{7c}$ for a purely ASCII narrative under UTF-8 — a ratio of $2^{9c}$, or approximately $2^{1350}$ for a 150-character story. Importantly, this advantage applies uniformly to all scripts: a Chinese-language narrative and an English-language narrative of the same length contribute the same raw symbol space, which reflects the script-neutral design of STORY2.

### 4.3. Guessing Entropy of a Narrative Key

The raw symbol space $2^{16c}$ is an upper bound on the guessing difficulty; the actual difficulty is governed by the guessing entropy of natural-language narratives, which is substantially lower due to the statistical structure of language. Shannon [29] estimated the entropy of written English at approximately 1.0–1.3 bits per character for a sophisticated predictor with access to long-range linguistic context. Under this estimate, a 150-character narrative provides a stranger-adversary

guessing entropy in the range

$$H \in [150,\ 195] \text{ bits},$$

placing it above the 128-bit security target even at the Shannon lower bound.

However, two factors raise the effective entropy above the Shannon estimate for general English. First, personal narratives contain private facts (names, places, events, secrets) that are not predictable from language statistics alone. An adversary who can model the statistics of English text cannot reconstruct a narrative that references, for example, a specific person's full name or a private experience. Second, natural writing variation — dialectal spelling, informal abbreviations, personal punctuation habits, and typographical idiosyncrasies — introduces additional unpredictability that neither frequency-based nor neural language models can fully capture [7]. The English language alone has over 170,000 recorded words, of which approximately 10,000 are in common daily use, distributed across national varieties (British, American, Australian) that differ in spelling, vocabulary, and idiom. A narrative that draws on personal experience and idiomatic expression therefore resists any word-list or corpus-based guessing strategy.

By contrast, conventional password-based systems treat the user secret as a short, low-entropy string and compensate with computationally expensive key-derivation functions such as PBKDF2 [13] or Argon2 [5]. STORY2 does not require memory hardness because the narrative itself provides sufficient guessing entropy: if $H \geq 128$ bits, the expected number of guesses required to recover the narrative exceeds $2^{127}$ regardless of how fast each guess can be evaluated.

### 4.4. Security Conditions

The analysis above is conditional. The following properties are required for the entropy estimates to hold:

1. The narrative must contain at least one private fact that is not known to, and cannot be inferred by, the adversary.
2. The narrative must not be copied from any published text, generated by an automated system, or shared with any third party.
3. The narrative must be at least 30 words in length, ensuring $H_{\text{stranger}} \geq 128$ bits under the Shannon lower bound.
4. The narrative must be entered exactly as originally composed, including any non-standard spelling, punctuation, or capitalisation, since the SHAKE-256 input is byte-exact.

A narrative that satisfies these conditions provides a key whose guessing entropy exceeds the 128-bit security target against a stranger adversary, and whose resistance to an insider adversary is bounded by the specificity of the private facts it contains.

## 5. Private S-Box Pool as a Secondary Security Layer

STORY2 is designed around a two-factor security model. The primary factor is the story key, which derives all round keys, the whitening key, and the S-box selection index. The secondary factor is the S-box pool itself. This section explains why a private pool was introduced, how it constitutes a genuine second security layer, and under what conditions that layer holds.

### 5.1. Motivation: The Story-Leak Residual Risk

A story key is a human-authored narrative. As established in Section 4, the guessing entropy of a well-constructed narrative exceeds $2^{128}$ bits against a stranger adversary and remains strong against a contextual adversary provided the narrative contains private facts. However, the possibility of key disclosure through non-cryptographic means — social engineering, coercion, physical access, or deliberate sharing followed by a change of trust — cannot be eliminated by cipher design alone.

In a conventional block cipher with a fixed public S-box, the non-linearity layer is fully known to every adversary before any ciphertext is produced. If the story key is subsequently recovered, the attacker immediately possesses every cipher parameter: the round keys, the whitening key, and the S-box. Decryption follows at negligible cost.

STORY2 addresses this residual risk by separating the non-linearity layer from the key derivation pipeline. The S-box pool is not derived from the story; it is an independently held secret shared between the communicating parties. Recovery of the story key therefore yields the round keys and the S-box selection *index*, but not the S-box *contents* unless the pool is also known.

### 5.2. Formal Definition and Pool Structure

**Definition 5.1** (S-Box Pool)**.** *Let $\mathcal{S} = \{S_0, S_1, \ldots, S_{|\mathcal{S}|-1}\}$ be a finite ordered collection of bijections $S_i : \{0, \ldots, 255\} \to \{0, \ldots, 255\}$, each independently validated as a permutation of all 256 byte values. Given $K_{\text{enc}} \in \{0,1\}^{256}$, the active S-box index is*

$$idx \;=\; \Big\lfloor \text{SHAKE-256}\big(\texttt{"story\_v2\_sbox//"} \,\|\, K_{\text{enc}},\ \ell = 2\big) \Big\rfloor_{\text{BE16}} \bmod |\mathcal{S}|. \tag{3}$$

*The pool $\mathcal{S}$ is either the built-in public pool or a privately constructed pool shared exclusively between the communicating parties.*

The critical architectural property is independence: $\mathcal{S}$ is not derived from the story key and cannot be computed from it. The story determines *which* S-box to use; it does not determine *what* the S-box contains.

### 5.3. The Secondary Security Layer

**Proposition 5.1** (Pool Secrecy as a Second Factor)**.** *Let $\mathcal{S}$ be a private pool unknown to the adversary. An adversary who recovers $K_{\text{enc}}$ can compute idx via Equation* (3) *but cannot determine the contents of $S_{idx}$ without access to $\mathcal{S}$. Decryption of any ciphertext encrypted under $\mathcal{S}$ therefore requires both $K_{\text{enc}}$ and $\mathcal{S}$.*

*Proof sketch. idx* is a function of $K_{\text{enc}}$ alone (Equation (3)). Given *idx*, the adversary knows which position in $\mathcal{S}$ contains the active S-box, but the bijection at that position is an element of the pool, which is not derivable from any public information or from the story key. Recovering $S_{idx}$ without the pool requires finding a specific bijection among all possible bijections on 256 elements. The total number of bijections on 256 elements is $256! \approx 2^{1684}$. Without additional information, the adversary must search this space, giving a brute-force cost of $2^{1684}$ trials. □

### 5.4. Why Relaxed DDT and LAT Do Not Weaken the Layer

A natural concern is whether custom S-boxes with relaxed differential or linear properties — for example DDT maximum of 6 or 8, and LAT maximum of 18, 20, or 22 — weaken the cipher's security when used in a private pool. The answer requires careful attention to the order of attacks.

Under a private pool, an adversary who wishes to mount a differential attack must first determine the active S-box's Difference Distribution Table. Computing the DDT of $S_{idx}$ requires knowing $S_{idx}$, which requires recovering the pool. Pool recovery costs $2^{1684}$ as established above. Only after the pool is recovered can the adversary exploit any DDT or LAT property.

The total attack cost is therefore the sum of two sequential steps:

$$\mathcal{C}_{\text{total}} \;=\; \underbrace{2^{1684}}_{\text{pool recovery}} \;+\; \underbrace{\left(\frac{\text{DDT}_{\max}}{256}\right)^{T_{\min}(r)}}_{\text{differential after recovery}} \;\approx\; 2^{1684}, \tag{4}$$

since the pool recovery term completely dominates. To illustrate concretely: even if a custom S-box has $\text{DDT}_{\max} = 8$, the best five-round differential probability is $(8/256)^{65} = 2^{-325}$, which still vastly exceeds any practical attack threshold. The two-step cost is $2^{1684} + 2^{-325} \approx 2^{1684}$.

Consequently, within a private pool, S-boxes with DDT values of 6 or 8 and LAT values of 18, 20, or 22 remain fully secure under the five-round construction. The $2^{1684}$ prerequisite makes the individual S-box quality cryptographically irrelevant to a passive adversary.

**Remark 5.1.** *This argument holds only for the passive adversary who cannot query the encryption system after recovering the story key. An active adversary with continued oracle access to the encryption function can recover any unknown 8-bit S-box using approximately* $256 \times 256 = 65{,}536$ *known-plaintext queries, a well-known result in the cryptanalysis of secret-S-box ciphers. The two-factor model therefore protects against story-key disclosure through non-cryptographic means*

*(social engineering, coercion, disclosure) but not against an adversary who retains active access to the encrypting system. This boundary must be understood by deployments that rely on pool secrecy as a security layer.*

### 5.5. Pool Secrecy Requirements

The secondary security layer holds if and only if the following conditions are satisfied.

1. **The pool must never be shared with any party other than the intended recipient.** Disclosure of the pool to any third party — including the system or application used for encryption — immediately reduces the construction to single-factor security. The pool is a shared secret of equal standing to the story key itself.

2. **The pool must not be derivable from any public information.** A pool generated by a publicly known algorithm seeded with a publicly known value provides no additional secrecy. The pool entries must be constructed and distributed through a channel independent of the cipher and the story key.

3. **The encrypting system must not be accessible to the adversary after a story-key compromise.** As noted above, oracle access to the encryption function reduces pool recovery to 65,536 known-plaintext queries. If the story key is compromised, the encrypting system must be taken offline immediately to preserve pool secrecy.

4. **The pool must be replaced after any suspected disclosure.** Because the pool is an independently held secret, it can be rotated without changing the story key, and vice versa. Regular pool rotation limits the window of exposure in the event of a partial compromise.

### 5.6. Default Pool and Deployment Modes

STORY2 ships with a built-in default pool whose contents are public. Against the public pool the secondary security layer does not apply: every adversary knows the pool, so recovery of the story key is sufficient for decryption. Security in default-pool mode rests entirely on the story key.

The design deliberately encourages users to replace the default pool with a privately constructed one. No modification to the source code is required; placing a `customju/sboxes.json` file alongside the module suffices. The pool format accepts any bijection on 256 elements, placing no constraint on the construction method beyond the mandatory bijection check. Users may construct pool entries by any means they consider appropriate, provided the results are validated bijections.

Table 3 summarises the security properties under each deployment mode.

**Table 3.** Security properties by S-box pool deployment mode.

| Pool mode | Story-key leak | S-box search cost |
|---|---|---|
| Default (public) | Decryption possible | 0 (pool is known) |
| Private (unknown) | Decryption blocked | $2^{1684}$ (passive) |
| Private (unknown) | Oracle access exists | $2^{16}$ (active) |

### 5.7. Novelty of the Construction

To our knowledge, no published block cipher employs an independently held, privately shared S-box pool as a second authentication factor. Existing designs with key-dependent S-boxes — most notably Blowfish [27] — derive their non-linearity layer directly from the encryption key. Compromise of the key in those designs immediately yields the S-box. STORY2's pool architecture breaks this coupling: the story key and the pool are orthogonal secrets, and neither can be derived from the other.

The construction is most accurately characterised as applying the two-factor authentication principle at the level of the cipher's non-linearity layer: decryption requires something you *know* (the story) and something you *hold* (the pool). This framing distinguishes the design from security through obscurity: the algorithm, the pool format, and the selection mechanism are all fully public; only the pool contents are secret, in the same sense that a private key is secret while the RSA algorithm is public.

## 6. Formal Security Analysis

This section presents the sixteen security results for STORY2 that are **mathematically proven or exactly computed**, as distinct from the empirical results reported in the Appendix (Section 11). The cipher operates over GF($2^8$) with irreducible polynomial $p(x) = x^8+x^4+x^3+x+1$. Block size is 128 bits; the 256-bit encryption key is derived from a Unicode passphrase via HMAC-SHA256; rounds are fixed at $r = 5$ with a final whitening ARK. All computations were performed by `formal.py` v3.0 on the production cipher parameters.

### 6.1. S-Box Cryptographic Properties

The S-box is selected deterministically from a pool of 255 validated bijections via SHAKE-256 keyed on `enc_key`. The following four properties are computed exactly for the selected S-box.

**Differential uniformity (DDT).** The Difference Distribution Table entry is

$$\mathrm{DDT}[\Delta x][\Delta y] = \#\big\{\, x \in \mathrm{GF}(2^8) : S(x) \oplus S(x \oplus \Delta x) = \Delta y \big\}.$$

Exhaustive computation over all 65,535 non-zero $(\Delta x, \Delta y)$ pairs yields $\mathrm{DDT}_{\max} = 4$, giving differential probability $\mathrm{DP} = 4/256 = 2^{-6}$ (Patterson–Wiedemann lower bound for $\mathrm{NL} \geq 112$).

**Nonlinearity (Walsh–Hadamard Transform).**

$$\mathrm{NL}(S) = \min_{\alpha \neq 0,\, \beta \neq 0} \left| \#\{x : \pi(x \cdot \alpha) = \pi(S(x) \cdot \beta)\} - 128 \right| = 128 - \max\text{-bias}.$$

The WHT is applied to each of the 8 output bits (complexity $O(n \cdot 2^n)$), giving NL = 112, max LAT bias = 16, correlation $\varepsilon = 0.125$. The theoretical optimum for an 8-bit bijection is NL = 120.

**Boomerang Connectivity Table (BCT) [12].**

$$\mathrm{BCT}[a][b] = \#\big\{x : S^{-1}(S(x) \oplus b) \oplus S^{-1}(S(x \oplus a) \oplus b) = a\big\}.$$

Exhaustive computation gives boomerang uniformity $\mathrm{BCT}_{\max} = 6$.

**Algebraic degree (Möbius / ANF transform).** The Algebraic Normal Form is computed via the Möbius subset-sum butterfly ($O(8 \times 2^8)$) independently for each output bit. All 8 output bits reach **degree 7**, the theoretical maximum for an 8-bit bijection.

**Table 4.** S-box cryptographic properties computed by exhaustive evaluation.

| **Property** | **Method** | **Value** | **Status** |
|---|---|---|---|
| $\mathrm{DDT}_{\max}$ (diff. uniformity) | Exhaustive, 65,535 pairs | 4 | Optimal |
| NL (nonlinearity) | Walsh–Hadamard Transform | 112 | Excellent |
| $\mathrm{BCT}_{\max}$ (boomerang unif.) | BCT exhaustive | 6 | Good |
| Algebraic degree | Möbius / ANF transform | 7 | Optimal (max) |

### 6.2. MDS Matrix Branch Number

STORY2 uses a 16×16 Cauchy MDS matrix $M[i][j] = \mathrm{GF\_inv}(x_i \oplus y_j)$, where $\{x_0, \ldots, x_{15}\}$ and $\{y_0, \ldots, y_{15}\}$ are disjoint subsets of $\mathrm{GF}(2^8)$. The branch number is $\mathrm{BN}(M) = \min_{v \neq 0}[\mathrm{wt}(v) + \mathrm{wt}(M \cdot v)]$.

**Cauchy algebraic proof.** The Cauchy determinant theorem guarantees that every $k \times k$ submatrix of $M$ is invertible over $\mathrm{GF}(2^8)$ for $1 \leq k \leq 16$, which implies $\mathrm{wt}(M \cdot v) \geq 17 - \mathrm{wt}(v)$ for all $v \neq 0$, and therefore:

$$\mathrm{wt}(v) + \mathrm{wt}(M \cdot v) \;\geq\; 17 \quad \forall\, v \neq 0 \qquad \therefore\ \mathrm{BN}(M) = 17.$$

This is a complete algebraic proof, not an estimate. Exhaustive verification over all 4,080 weight-1 inputs (16 positions × 255 non-zero values) and 5,000 random weight-2 inputs confirms BN = 17. This equals the Singleton bound $n + 1 = 17$, the theoretical maximum for any 16×16 MDS matrix.

### 6.3. Differential, Linear, and Boomerang Security Bounds

**Differential bound.** Any $r$-round trail begins with $\geq 1$ active byte. Since BN = 17, any weight-1 input to the MDS layer forces all 16 output bytes active in a single pass. The minimum active S-box count is therefore:

$$T_{\min}(r) = 1 + 16(r-1), \qquad \mathrm{DP}_{\max}(r) \;\leq\; \left(\frac{4}{256}\right)^{T_{\min}(r)}.$$

**Table 5.** Per-round security bounds for STORY2 (5-round configuration in bold).

| Rounds | Min. active S-boxes | Diff. security | Lin. security (LC) | Status |
|---|---|---|---|---|
| 3 | 33 | 132 bits | 198 bits | $\geq 128$ |
| 4 | 49 | 198 bits | 294 bits | $\geq 128$ |
| **5** | **65** | **390 bits** | **262 bits** | $\geq 128$ |

**MILP confirmation.** The minimum active S-box count was independently verified by Mixed Integer Linear Programming (PuLP/CBC solver [19]). Both differential and linear MILP models return $T_{\min} = 65$ at 5 rounds, confirming the direct formula exactly.

**Linear bound (Matsui's piling-up lemma [18]).** For $t$ independent S-box approximations each with correlation $\varepsilon = 0.125$:

$$|\varepsilon_{\text{total}}| = 2^{t-1} \cdot \varepsilon^t = 2^{64} \cdot 0.125^{65} \approx 2^{-131}, \qquad \text{LC security} = -2\log_2 |\varepsilon_{\text{total}}| = 262 \text{ bits}.$$

**Boomerang bound (Wagner [31] + BCT).** Splitting $E = E_1 \circ E_0$ with 2 rounds in $E_0$ and 3 in $E_1$:

$$\begin{aligned} p_0 &= \mathrm{DP}_{\max}(2) = (4/256)^{17} \approx 2^{-102}, \\ p_1 &= \mathrm{DP}_{\max}(3) = (4/256)^{33} \approx 2^{-198}, \\ p_{\text{boom}} &= (p_0 \cdot p_1)^2 = 2^{-600}. \end{aligned}$$

Boomerang security: 600 bits.

### 6.4. Algebraic Properties

**Degree growth and division property.** The MDS layer is linear and degree-preserving; only SubBytes multiplies the algebraic degree. The cipher reaches degree saturation at round 3:

$$\deg_r = \min(7^r,\, 128), \qquad \deg_3 = \min(343, 128) = 128 \;\bigstar \text{ Saturated.}$$

From round 3 onward all 128 output bit functions are maximally non-linear and balanced, blocking integral attacks. The S-box satisfies $\bigoplus_{x=0}^{255} S(x) = 0$ (verified by direct computation), blocking one-round integral distinguishers.

**Impossible differential resistance.** With BN = 17, any weight-$k$ input produces weight-16 output in one round. For any split ($f$ forward, $b = r - f$ backward rounds):

$$w_f + w_b = 16 + 16 = 32 > 16 \quad \text{for all } 1 \le f \le r - 1.$$

Every miss-in-the-middle pair produces a contradiction, proving no full-round impossible differential structural path exists.

### 6.5. Key Schedule and AEAD Security

**Key schedule.** All derived values use HMAC-SHA256 with distinct domain labels, ensuring computational independence under the HMAC PRF model [2]:

**Table 6.** STORY2 key schedule derivation and independence properties.

| Value | Domain prefix | Size | Property |
|---|---|---|---|
| `enc_key`, `mac_key` | `enc/mac\|\|story_v2_master` | 32 B | Independent PRF |
| Round keys ($5 \times 16$) | `story_v2_keys` | 80 B | Preimage $2^{256}$ |
| `whitening_key` | `story_v2_whitening` | 16 B | Independent |
| S-box index | `story_v2_sbox` | Index | Key-dependent |

All 5 round keys are verified distinct. Recovering `enc_key` from any round key requires inverting HMAC-SHA256 (work $2^{256}$). Slide attacks are blocked by design.

**AEAD security (Encrypt-then-MAC).** STORY2 uses the Encrypt-then-MAC (EtM) construction:

$$C = \text{STORY2-CTR}(pt,\, nonce,\, enc\_key), \qquad tag = \text{HMAC-SHA256}(mac\_key,\, nonce \| C).$$

The IND-CPA advantage satisfies $\varepsilon_{\text{CPA}} \le q^2/2^{64} + qL/2^{128}$. HMAC-SHA256 provides INT-CTXT with forgery probability $\le q/2^{256}$. By the Encrypt-then-MAC composition theorem [4], the full construction achieves IND-CCA2.

### 6.6. Summary of 16 Proven Results

**Table 7.** All 16 mathematically proven security results for STORY2.

| # | Property | Method | Value | Result |
|---|---|---|---|---|
| 1 | $DDT_{max}$ (diff. uniformity) | Exhaustive computation | 4 | Optimal |
| 2 | Nonlinearity NL | Walsh–Hadamard Transform | 112 | Excellent |
| 3 | $BCT_{max}$ (boomerang unif.) | BCT exhaustive | 6 | Good |
| 4 | Algebraic degree | Möbius / ANF transform | 7 | Optimal |
| 5 | Branch number BN | Cauchy proof + exhaustive | 17 | Optimal |
| 6 | Differential security | Active S-box formula | 390 bits | Secure |
| 7 | MILP differential trail | MILP / CBC solver | 65 active | Confirmed |
| 8 | MILP linear trail | MILP / CBC solver | 65 active | Confirmed |
| 9 | Degree saturation | ANF growth model | Round 3 | Saturated |
| 10 | Integral resistance | XOR-sum balance | 0 (balanced) | Secure |
| 11 | Impossible differentials | Miss-in-the-middle | BN= 17 $\Rightarrow$ all $w_f+w_b>16$ | Proven |
| 12 | Slide/symmetry absence | Round key distinctness | All 5 unique | Proven |
| 13 | Linear security (LC) | Matsui piling-up lemma | 262 bits | Secure |
| 14 | Boomerang security | Wagner / BCT formula | 600 bits | Secure |
| 15 | Key schedule | HMAC preimage + PRF | $2^{256}$ security | Secure |
| 16 | AEAD security | EtM composition theorem | IND-CCA2 + INT-CTXT | Proven |

*All computations performed by* `formal.py` *v3.0 on* STORY2 *v0.4.0 with 5 rounds. S-box selected for story key "JuCrypt was made with love, not to compete against existing ciphers.", using the default S-box pool.*

# 7. The Cauchy MDS Matrix

## 7.1. State Representation

The STORY2 block is 16 bytes, treated as a column vector $\mathbf{s} = [s_0, s_1, \ldots, s_{15}]^\top$ where each $s_i \in \{0, \ldots, 255\}$. The MIX step multiplies this vector by a $16 \times 16$ matrix $M$ over $\mathbb{GF}_{2^8}$:

$$\text{output}[i] \;=\; \bigoplus_{j=0}^{15} M[i][j] \cdot s_j, \qquad i = 0, \ldots, 15, \tag{5}$$

where $\cdot$ is $\mathbb{GF}_{2^8}$ multiplication and $\oplus$ is XOR.

## 7.2. Constructing the Cauchy Matrix

A Cauchy matrix is guaranteed invertible and MDS by construction, provided its defining sequences satisfy the following conditions.

### *7.2.1. Row and Column Sequences*

STORY2 generates two sequences from successive powers of $\alpha = \texttt{0x02}$ in $\mathbb{GF}_{2^8}$:

$$x_i = \alpha^i, \quad i = 0, \ldots, 15, \tag{6}$$

$$y_j = \alpha^{j+16}, \quad j = 0, \ldots, 15. \tag{7}$$

The element $\alpha = \texttt{0x02}$ has multiplicative order 51 in $\mathbb{GF}_{2^8}$ under the irreducible polynomial (1). Since only 32 consecutive powers (indices 0 through 31) are required and $32 < 51$, all 32 values are distinct. The index ranges $[0, 15]$ and $[16, 31]$ are disjoint, so no value appears in both sequences. Therefore $x_i \oplus y_j \neq 0$ for all $i, j$, and the GF inverse in the Cauchy formula below is always well-defined.

### *7.2.2. The Cauchy Entry Formula*

Every entry of the matrix is

$$\boxed{M[i][j] \;=\; \big(x_i \oplus y_j\big)^{-1},} \tag{8}$$

where $(\cdot)^{-1}$ denotes the $\mathbb{GF}_{2^8}$ multiplicative inverse.

## 7.3. Invertibility of the Cauchy Matrix

**Theorem 7.1** (Cauchy Invertibility)**.** *A Cauchy matrix over any field $\mathbb{F}$ is invertible if and only if all $x_i$ are distinct, all $y_j$ are distinct, and $\{x_0, \ldots, x_{n-1}\} \cap \{y_0, \ldots, y_{n-1}\} = \emptyset$.*

*Proof.* The Cauchy determinant formula gives

$$\det(C) \;=\; \frac{\prod_{i>j}(x_i - x_j)\,\prod_{i>j}(y_j - y_i)}{\prod_{i,j}(x_i - y_j)}. \tag{9}$$

Under the stated conditions every factor in the numerator is non-zero and every factor in the denominator is non-zero, so $\det(C) \neq 0$ and $C$ is invertible. □ □

### 7.4. Branch Number of the STORY2 Mix

**Theorem 7.2** (Branch Number)**.** *The matrix $M$ defined by* (8) *is MDS over $\mathbb{GF}_{2^8}$ with $\mathcal{B}(M) = 17$, the theoretical maximum for any $16 \times 16$ linear map over $\mathbb{GF}_{2^8}$.*

*Proof sketch.* A Cauchy matrix is MDS because every square submatrix of a Cauchy matrix is itself Cauchy and therefore invertible by Theorem 7.1. Invertibility of all square submatrices implies that the associated linear code meets the Singleton bound, which is equivalent to $\mathcal{B}(M) = n + 1 = 17$. The branch number was additionally confirmed by exhaustive enumeration over all 4,080 weight-1 inputs (16 positions $\times$ 255 non-zero values) and 5,000 randomly sampled weight-2 inputs, all yielding $\mathrm{wt}(\mathbf{x}) + \mathrm{wt}(M\mathbf{x}) \geq 17$. □

### 7.5. Why the Permutation Layer Was Removed

Standard SPN designs include a byte permutation alongside the MDS diffusion layer to ensure that bytes from different columns interact across rounds. Without such a permutation, a column-restricted MDS matrix would not propagate differences across the full state, and the full-state branch number would remain bounded by the column width.

STORY2's $16 \times 16$ MDS matrix operates over all 16 bytes simultaneously. Since $\mathcal{B}(M) = 17$, any weight-1 input already activates all 16 output bytes in a single pass. A permutation layer rearranges active bytes but cannot increase their count beyond what the MDS layer already guarantees. It is therefore cryptographically redundant and has been eliminated. This reduces the round function to three operations — ARK, SubBytes, and Mix — whilst preserving the full-state diffusion property.

### 7.6. Python Implementation

```
def _derive_cauchy_matrix():
    gf_inv = [0] * 256
    for a in range(1, 256):
        for b in range(1, 256):
            if _GF[a][b] == 1:
                gf_inv[a] = b; break
    def pow_alpha(n):
```

```
        a = 1
        for _ in range(n): a = _GF[a][2]
        return a
    xs = [pow_alpha(i)      for i in range(16)]
    ys = [pow_alpha(i + 16) for i in range(16)]
    return [[gf_inv[xs[i] ^ ys[j]] for j in range(16)]
            for i in range(16)]
_MDS_MATRIX = _derive_cauchy_matrix()
```

**Listing 1.** Cauchy MDS matrix construction (`story2.py`).

```
@staticmethod
def _mix(state):
    result = [0] * 16
    for i in range(16):
        acc = 0
        row = _MDS_MATRIX[i]
        for j in range(16):
            acc ^= _GF[row[j]][state[j]]
        result[i] = acc
    return result
```

**Listing 2.** MDS mix step (`story2.py`).

The C acceleration layer additionally precomputes a $16 \times 16 \times 256$ table `MDS_FLAT[i][j][v] = GF[MDS[i][j]][v]`, reducing the inner loop of Mix to 16 XOR operations with no runtime multiplication.

## 8. Cipher Design

### 8.1. Design Philosophy

Story2 is built on three principles that distinguish it from conventional SPN designs.

*Narrative as key.* The story string is the direct source of every cipher parameter. Any change to any character produces a completely different S-box index and round key set. The cipher makes no attempt to compensate for a weak story; the strength of the key material is the responsibility of the user.

*Full parameterisation.* Every component of the substitution layer is replaceable via the two-level S-box pool architecture described in Section 5.

*Openness of structure.* Security rests entirely on the secrecy of the story string, not on any concealment of the algorithm. This follows directly from Kerckhoffs's principle [14].

### 8.2. Key Derivation

The story string $s$ is first encoded as UTF-16-LE bytes. The derivation pipeline then proceeds in two steps:

$$prk = \textsc{Hmac-Sha256}\big(\texttt{"story\_v2\_salt"},\ s_{\text{UTF-16-LE}}\big) \tag{10}$$

$$K_{\text{enc}} = \textsc{Hmac-Sha256}\big(prk,\ \texttt{"enc||story\_v2\_master\textbackslash x01"}\big) \tag{11}$$

$$K_{\text{mac}} = \textsc{Hmac-Sha256}\big(prk,\ \texttt{"mac||story\_v2\_master\textbackslash x02"}\big) \tag{12}$$

The first step applies HMAC-SHA256 with a fixed salt label to extract a 32-byte pseudorandom key `prk` from the variable-length story string. The second step expands `prk` into two independent 32-byte subkeys using domain-separated labels, following the HKDF-expand pattern [16]. The label separation guarantees $K_{\text{enc}} \perp K_{\text{mac}}$, which is a necessary condition for the security of the Encrypt-then-MAC composition [3].

### 8.3. S-Box Selection

Story2 selects one S-box from the active pool deterministically from $K_{\text{enc}}$:

$$idx \ = \ \Big\lfloor \textsc{Shake-256}\big(\texttt{"story\_v2\_sbox||"} \,\|\, K_{\text{enc}},\ \ell = 2\big) \Big\rfloor_{\text{BE16}} \bmod |\mathcal{S}|. \tag{13}$$

Every S-box in the pool is validated at load time to be a bijection: exactly 256 entries whose sorted values equal $\{0, 1, \ldots, 255\}$. The pool architecture supports a private custom pool as described in Section 5.

### 8.4. Fixed Round Architecture

Story2 uses a fixed round count of $r = 5$. This value is determined by the formal security analysis in Section 6: three rounds suffice for algebraic degree saturation and for differential security exceeding 128 bits; five rounds provide a safety margin of 390 bits differential security and 262 bits linear security. A fixed round count eliminates the need to transmit a per-message salt and simplifies the authenticated data scope.

### 8.5. Round Key Expansion

Five round keys of 16 bytes each are derived from $K_{\text{enc}}$ via SHAKE-256:

$$\big[k_0 \| k_1 \| k_2 \| k_3 \| k_4\big] \ = \ \textsc{Shake-256}\big(\texttt{"story\_v2\_keys||"} \,\|\, K_{\text{enc}},\ \ell = 80\big). \tag{14}$$

A separate 16-byte whitening key is derived independently:

$$k_{\text{white}} \ = \ \textsc{Shake-256}\big(\texttt{"story\_v2\_whitening||"} \,\|\, K_{\text{enc}},\ \ell = 16\big). \tag{15}$$

The whitening key is applied after the final round to close the last Mix layer against key-recovery attacks that exploit the known structure of the final mixing step.

### 8.6. The SPN Round Structure

Each of the five rounds executes three operations in the following order.

1. **AddRoundKey (ARK):** $s_i \leftarrow s_i \oplus k_r[i], \quad i = 0, \ldots, 15.$
2. **SubBytes:** $s_i \leftarrow S_{idx}[s_i]$, where $S_{idx}$ is the pool-selected S-box.
3. **StoryMix:** apply the $16 \times 16$ Cauchy MDS matrix (Equation (5)).

After the fifth round, a final whitening ARK is applied: $s_i \leftarrow s_i \oplus k_{\text{white}}[i]$.

SubBytes is the sole source of non-linearity. Without it, the full SPN reduces to a linear map over GF(2), which is vulnerable to Gaussian elimination with $n$ known plaintext-ciphertext pairs. The StoryMix layer is applied in the forward direction only; because STORY2 uses CTR mode, the inverse of $M$ is never required.

### 8.7. Counter Mode Encryption

A fresh 8-byte nonce $N$ is drawn from `os.urandom(8)` on every `encrypt()` call. For each 16-byte plaintext block at position $c = 0, 1, 2, \ldots$ the counter block and ciphertext block are:

$$\text{CB}(c) = N \,\|\, \langle c \rangle_8, \qquad C_c = P_c \oplus E_K\big(\text{CB}(c)\big), \tag{16}$$

where $\langle c \rangle_8$ is $c$ as an 8-byte big-endian integer and $E_K$ denotes the keyed block cipher. The 8-byte nonce supports up to $2^{64}$ distinct counter blocks per key, covering up to $2^{68}$ bytes of plaintext.

### 8.8. Encrypt-then-MAC Authentication

After CTR encryption the ciphertext is authenticated under $K_{\text{mac}}$:

$$\tau \;=\; \text{HMAC-SHA256}\big(K_{\text{mac}},\; N \,\|\, C\big). \tag{17}$$

The nonce and the full ciphertext are both covered by the tag, binding them into a single authenticated unit. The Encrypt-then-MAC construction is INT-CTXT and IND-CCA2 secure when $K_{\text{enc}} \perp K_{\text{mac}}$ [3], which is guaranteed by the domain-separated derivation in (12). Decryption verifies the tag using `hmac.compare_digest` before returning any plaintext byte, preventing timing-based tag forgery.

The ciphertext tuple returned by `encrypt()` is:

$$\big(\, C,\; N,\; \tau \,\big). \tag{18}$$

All three fields are required for decryption.

### 8.9. Nonce Reuse

CTR mode is not nonce-misuse resistant: two ciphertexts produced under the same $(K, N)$ pair satisfy $C_1 \oplus C_2 = P_1 \oplus P_2$. STORY2 prevents this by generating a fresh uniformly random 64-bit nonce on every encryption call. The birthday bound for a collision probability exceeding 50% is $2^{32} \approx 4 \times 10^9$ encryptions under the same story key, which represents a practical upper limit for deployment. Empirical tests confirm that nonce recovery is correct in all 2,000 trials and that MAC verification rejects every tampered ciphertext presented to it.

## 9. Limitations and Future Work

The following open problems represent independent research directions.

- **Minimum secure round count.** The formal analysis establishes that three rounds suffice for 128-bit differential and linear security under the default S-box pool. Whether three rounds constitutes a tight lower bound under all known attack families has not been proved; this remains an open question.
- **Private pool minimum size.** Section 5 establishes that a private pool of any size provides a $2^{1684}$ passive-adversary search bound. The minimum pool size such that an informed adversary (who knows the quality thresholds) cannot enumerate the pool faster than searching $2^{256}$ story keys has not been formally derived.
- **Oracle attack boundary for pool recovery.** An active adversary who retains query access after a story-key compromise can recover any 8-bit S-box in approximately 65,536 known-plaintext queries. Whether this bound can be tightened under STORY2's specific round structure and MDS diffusion has not been studied.
- **Hardware efficiency of the $16 \times 16$ MDS layer.** The software implementation uses a $16 \times 16 \times 256$ lookup table. The optimal trade-off between table size and throughput in a hardware implementation has not been analysed.
- **Independent cryptanalytic review.** Sixteen security properties have been computed by `formal.py` v3.0, but the cipher has not yet undergone independent public cryptanalysis. Results that identify weaknesses are of equal value to results that confirm strength; the authors invite independent analysis with no obligation of coordinated disclosure.

## 10. Implementation Considerations

The reference implementation is available on PyPI as part of the JuCrypt library (`pip install jucrypt`). The pure Python path is suitable for correctness testing and moderate workloads;

the C-accelerated path (`STORYC`) should be used for any performance-sensitive deployment. The $16 \times 16 \times 256$ MDS lookup table occupies approximately 65 KB in memory. Users constructing a custom S-box pool should independently verify the DDT and LAT of each pool entry before deployment, as the cipher validates bijection only and does not impose quality bounds on custom entries.

## 11. Conclusion

We have presented STORY2, a symmetric block cipher whose primary key is a human-authored narrative. The central design contributions are a $16 \times 16$ Cauchy MDS diffusion matrix that achieves the theoretical maximum branch number of 17 and enforces full 16-byte diffusion in a single pass, and a two-factor security model in which a privately held S-box pool provides an independent $2^{1684}$-strength secondary layer against a passive adversary who has recovered the story key.

The elimination of the permutation layer, made possible by the full-state MDS diffusion, simplifies the round function without weakening any security property. The formal analysis confirms sixteen proven security results covering all principal attack families, with differential security of 390 bits and boomerang security of 600 bits at five rounds.

The primary open question is not cryptographic but empirical: a human-generated narrative provides high guessing entropy against a stranger adversary, but that entropy depends on the author adhering to the conditions set out in Section 4. A cipher cannot enforce the quality of its own key material; this responsibility rests with the user. The guidelines in Section 4 are therefore as much a part of the design as the mathematics.

## Acknowledgements

The parent library of STORY2 is JuCrypt, an abbreviation of Ju Wenjun Cryptography. This work is dedicated to the Women's World Chess Champion Ju Wenjun, whose work we admire. The authors acknowledge the use of Grammarly for grammar checking and Claude (Anthropic) for assistance with citation formatting, code comments, and test result analysis. STORY2 is submitted without the backing of a formal institutional security review. Independent analysis from any researcher, at any level of expertise, is welcomed; any well-documented finding will be acknowledged with full attribution in future versions of this work.

## References

[1] Paulo S. L. M. Barreto and Vincent Rijmen. The Whirlpool hashing function. In *First Open NESSIE Workshop*, pages 1–24, Leuven, Belgium, 2003. NESSIE Consortium.

[2] Mihir Bellare. New proofs for NMAC and HMAC: Security without collision resistance. *CRYPTO*, 2006.

[3] Mihir Bellare and Chanathip Namprempre. Authenticated encryption: Relations among notions and analysis of the generic composition paradigm. In *Advances in Cryptology — ASIACRYPT 2000*, volume 1976 of *Lecture Notes in Computer Science*, pages 531–545. Springer, 2000.

[4] Mihir Bellare and Chanathip Namprempre. Authenticated encryption: Relations among notions and analysis of the generic composition paradigm. In *ASIACRYPT*, 2000.

[5] Alex Biryukov, Daniel Dinu, Dmitry Khovratovich, and Simon Josefsson. Argon2 memory-hard function for password hashing and proof-of-work applications. RFC 9106, Internet Engineering Task Force, 2021.

[6] Joseph Bonneau. The science of guessing: Analysing an anonymised corpus of 70 million passwords. In *2012 IEEE Symposium on Security and Privacy (SP)*, pages 538–552. IEEE, 2012.

[7] Joseph Bonneau and Ekaterina Shutova. Linguistic properties of multi-word passphrases. In *Financial Cryptography and Data Security, Workshop on Usable Security (USEC)*, 2012.

[8] Joan Daemen, Lars Knudsen, and Vincent Rijmen. The block cipher SHARK. In *Fast Software Encryption — FSE 1996, LNCS*, volume 1039, pages 99–111, Berlin, Germany, 1996. Springer.

[9] Joan Daemen and Vincent Rijmen. *The Design of Rijndael: AES — The Advanced Encryption Standard*. Springer, Berlin, Heidelberg, 2002.

[10] Jian Guo, Thomas Peyrin, and Axel Poschmann. The PHOTON family of lightweight hash functions. In *Advances in Cryptology — CRYPTO 2011, LNCS*, volume 6841, pages 222–239, Berlin, Germany, 2011. Springer.

[11] Kishan Chand Gupta, Sumit Kumar Pandey, Indranil Ghosh Ray, and Susanta Samanta. Cryptographically significant MDS matrices over finite fields: A brief survey and some generalised results. *Advances in Mathematics of Communications*, 13(4):779–843, 2019.

[12] Jérémy Jean et al. Cryptanalysis of SKINNY in the framework of the SKINNY-AEAD and SKINNY-Hash families. BCT introduced in: Kim, Bogdanov et al., EUROCRYPT 2019.

[13] B. Kaliski. PKCS #5: Password-based cryptography specification version 2.0. RFC 2898, Internet Engineering Task Force, 2000.

[14] Auguste Kerckhoffs. La cryptographie militaire. *Journal des Sciences Militaires*, 9:5–38, 1883.

[15] Hugo Krawczyk, Mihir Bellare, and Ran Canetti. HMAC: Keyed-hashing for message authentication. RFC 2104, Internet Engineering Task Force, 1997.

[16] Hugo Krawczyk and Pasi Eronen. HMAC-based extract-and-expand key derivation function (HKDF). RFC 5869, Internet Engineering Task Force, 2010.

[17] Tran Thi Luong, Nguyen Van Long, and Bac Vo. Efficient implementation of the linear layer of block ciphers with large MDS matrices based on a new lookup table technique. *PLOS ONE*, 19(6):e0304873, 2024.

[18] Mitsuru Matsui. Linear cryptanalysis method for DES cipher. In *Advances in Cryptology — EUROCRYPT 1993*, volume 765 of *Lecture Notes in Computer Science*, pages 386–397. Springer, 1994.

[19] Stuart Mitchell et al. PuLP: A linear programming toolkit for Python.

[20] M. Mousavi, M. Zaghian, and M. Esmaeili. Involutory-multiple-lightweight MDS matrices based on Cauchy-type matrices. *Advances in Mathematics of Communications*, 15(4):589–606, 2021.

[21] National Institute of Standards and Technology. SHA-3 standard: Permutation-based hash and extendable-output functions. Federal Information Processing Standards Publication 202, U.S. Department of Commerce, 2015.

[22] National Institute of Standards and Technology. SHA-3 standard: Permutation-based hash and extendable-output functions. Federal Information Processing Standards Publication 202, U.S. Department of Commerce, 2015.

[23] Peter Norvig. English letter frequency counts: Mayzner revisited. https://norvig.com/mayzner.html, 2012. Accessed March 2025.

[24] Marek Palatinus, Pavol Rusnak, Aaron Voisine, and Sean Bowe. BIP-39: Mnemonic code for generating deterministic keys. https://github.com/bitcoin/bips/blob/master/bip-0039.mediawiki, 2013. Bitcoin Improvement Proposal 39.

[25] Arnold G. Reinhold. Diceware passphrase home page. In *Diceware Passphrase*, 1995.

[26] Andrew Rukhin, Juan Soto, James Nechvatal, Miles Smid, Elaine Barker, Stefan Leigh, Mark Levenson, Mark Vangel, David Banks, Alan Heckert, James Dray, and San Vo. A statistical test suite for random and pseudorandom number generators for cryptographic applications. Special Publication NIST SP 800-22 Rev. 1a, National Institute of Standards and Technology, 2010.

[27] Bruce Schneier. Description of a new variable-length key, 64-bit block cipher (Blowfish). In *Fast Software Encryption — FSE 1993*, volume 809 of *Lecture Notes in Computer Science*, pages 191–204. Springer, 1994.

[28] Claude E. Shannon. Communication theory of secrecy systems. *Bell System Technical Journal*, 28(4):656–715, 1949.

[29] Claude E. Shannon. Prediction and entropy of printed english. *Bell System Technical Journal*, 30(1):50–64, 1951.

[30] Richard Shay, Saranga Komanduri, Adam L. Durity, Phillip Huh, Michelle L. Mazurek, Sean M. Segreti, Blase Ur, Lujo Bauer, Nicolas Christin, and Lorrie Faith Cranor. Correct horse battery staple: Exploring the usability of system-assigned passphrases. In *Proceedings of the 8th Symposium on Usable Privacy and Security (SOUPS 2012)*, pages 7:1–7:20, New York, NY, USA, 2012. ACM.

[31] David Wagner. The boomerang attack. In *Fast Software Encryption (FSE)*, 1999.

[32] A. F. Webster and Stafford E. Tavares. On the design of S-boxes. In *Advances in Cryptology — CRYPTO 1985*, volume 218 of *Lecture Notes in Computer Science*, pages 523–534. Springer, 1986.

## Appendix — Empirical Test Results

This appendix presents the results of the basic test suite applied to STORY2 across 2,000 story keys. The suite measures cipher correctness, diffusion quality, statistical uniformity, and implementation timing. It directly addresses Research Questions 1, 2, 4, and 6 from Section 1.4.

### Test Configuration

Two test runs were conducted on 30 March 2026.

**Cold run** (run ID 2724667016, timestamp 05:40:55 UTC): no CPU or cache warmup prior to measurement. This represents a first-call scenario from a fresh process.

**Warm run** (run ID 2039078462, timestamp 05:54:31 UTC): executed immediately after the cold run on the same process, so CPU caches and OS scheduler state were already conditioned. This represents a sustained-use scenario.

Both runs used the C-accelerated implementation (`STORYC`), a fixed plaintext length of 44 bytes, and 2,000 story keys spanning story lengths of 14 to 280 characters. Keys were drawn from five distinct scripts: Latin, Arabic, Cyrillic, Vietnamese, and Chinese (CJK), confirming script-neutral operation under UTF-16-LE encoding. Specific hardware and operating system details were not recorded at the time of testing; this is acknowledged as a limitation of the current test

infrastructure. Full results are available in the accompanying data files `story_basic_final.csv` (cold) and `story_basic_final_re.csv` (warm).

### Representative Sample

Table 8 presents eight representative rows drawn from the cold run, ordered by story length, to illustrate performance across the tested key range. All aggregate statistics in the Aggregate Results subsection below are computed from the full 2,000-row datasets.

**Table 8.** Eight representative cold-run results (of 2,000 total), ordered by story length.

| Len | Story (excerpt) | Ava % | SAC | KS % | $H$ (b/B) | $\chi^2$ | NIST | DDT | LAT | $t$ (ms) |
|---|---|---|---|---|---|---|---|---|---|---|
| 14 | Nadia saw *bahr* (Arabic). | 50.476 | 0.5048 | 50.370 | 7.9911 | P | F | 4 | 16 | 0.134 |
| 27 | Elena saw estrella perdida. | 49.744 | 0.4974 | 49.930 | 7.9908 | P | P | 4 | 16 | 0.140 |
| 35 | Pierre walked through forêt… | 49.432 | 0.4943 | 50.137 | 7.9917 | P | P | 4 | 16 | 0.135 |
| 46 | The last letter never arrived… | 49.902 | 0.4990 | 50.424 | 7.9930 | P | P | 4 | 16 | 0.132 |
| 63 | Natuto syang magpaalam… | 50.287 | 0.5029 | 49.646 | 7.9921 | P | P | 4 | 16 | 0.138 |
| 76 | *Yeda byla gotova…* (Cyrillic) | 49.872 | 0.4987 | 50.085 | 7.9917 | P | P | 4 | 16 | 0.134 |
| 92 | *On pisal mnogo stikhov…* (Cyrillic) | 50.330 | 0.5033 | 50.302 | 7.9921 | P | P | 4 | 16 | 0.134 |
| 118 | Anh ãy nuôi mọt chạu tre… | 50.122 | 0.5012 | 50.293 | 7.9931 | P | P | 4 | 16 | 0.135 |

Column headers: Len = story length (characters); Ava = avalanche effect; SAC = mean Strict Avalanche Criterion; KS = key sensitivity; $H$ = Shannon entropy (bits/byte); $\chi^2$/NIST = pass (P) or fail (F); DDT/LAT = S-box maximum values; $t$ = median encrypt latency. Ideal values: Ava = 50%, SAC = 0.500, KS = 50%.

### Aggregate Results

Table 9 presents the aggregate statistics for both runs across all 2,000 story keys.
Ideal values are shown in parentheses where applicable. Cold = no CPU or cache warmup; Warm = immediate re-run on the same process.

### Discussion

**Correctness.** All 2,000 encrypt-decrypt roundtrips succeeded in both runs, confirming that the CTR-mode keystream is reproduced exactly and that the authenticated decryption path accepts all correctly formed tuples (18).

**Table 9.** Aggregate statistics across 2,000 story keys (cold and warm runs).

| Metric | Cold run | Warm run |
|---|---|---|
| Roundtrip correct | 2000/2000 (100%) | 2000/2000 (100%) |
| Avalanche mean (%) | 50.003 (50.000) | 50.007 (50.000) |
| Avalanche std (%) | 0.393 | 0.388 |
| Avalanche range (%) | [48.694, 51.477] | [48.810, 51.331] |
| SAC mean | 0.50003 (0.5000) | 0.50007 (0.5000) |
| SAC std | 0.003925 | 0.003884 |
| BIC pass ($< 0.10$ threshold) | 2000/2000 (100%) | 2000/2000 (100%) |
| BIC max corr (mean / max) | 0.0310 / 0.0429 | 0.0310 / 0.0442 |
| Key sensitivity mean (%) | 50.013 (50.000) | 49.991 (50.000) |
| Shannon entropy mean (bits/B) | 7.9916 (8.000) | 7.9916 (8.000) |
| Shannon entropy min (bits/B) | 7.9890 | 7.9890 |
| $\chi^2$ pass ($\alpha = 0.05$) | 1898/2000 (94.9%) | 1901/2000 (95.0%) |
| NIST monobit pass ($\alpha = 0.01$) | 1974/2000 (98.7%) | 1986/2000 (99.3%) |
| $\mathrm{DDT}_{\max}$ (all keys) | 4 | 4 |
| $\mathrm{LAT}_{\max}$ (all keys) | 16 | 16 |
| Serial corr lag-1 max $\lvert r \rvert$ | 0.0274 | 0.0233 |
| Timing: median latency (ns) | 135,100 | 135,400 |
| Timing: CV $< 5\%$ | 1928/2000 (96.4%) | 1894/2000 (94.7%) |

**Avalanche and SAC (RQ2).** The mean avalanche effect of 50.003% (cold) and 50.007% (warm) is within 0.004 percentage points of the theoretical ideal of 50%, and the per-run standard deviation of 0.39% is consistent with statistical sampling noise across a finite key set. The strict Avalanche Criterion is satisfied by the Cauchy MDS matrix in a single round (Section 7.4); the per-key variation in the above figures arises from the interaction of the key-dependent S-box and the fixed-round structure, not from diffusion failure. The mean and standard deviation are stable across the cold and warm runs (delta of 0.004%), confirming that the result is reproducible and independent of CPU state.

**Bit Independence Criterion.** All 2,000 keys pass the BIC in both runs. The mean maximum inter-output-bit correlation is 0.031, well below the pass threshold of 0.10 (which is three times the null expectation of $\approx 0.033$ under independence, as noted in the test suite source). This confirms that individual output bits do not carry correlated information about each other across the tested key population.

**Key sensitivity.** The mean key sensitivity of 50.01% (cold) and 49.99% (warm) shows that a single-bit change in the story key changes approximately half of all ciphertext bits in expectation, independent of which story key is in use. The standard deviation of 0.27% indicates consistent behaviour across all 2,000 keys.

**Statistical uniformity.** The mean Shannon entropy of 7.9916 bits per byte is close to the maximum of 8.0, and the minimum observed value of 7.989 bits per byte across all keys confirms that no key produces systematically non-uniform output. The $\chi^2$ pass rate of 94.9% (cold) and 95.0% (warm) is within one percentage point of the expected value of 95.0% at significance level $\alpha = 0.05$: 102 cold-run failures against an expectation of 100. The NIST monobit pass rates of 98.7% (cold) and 99.3% (warm) are consistent with an expected failure rate of 1% at $\alpha = 0.01$ (26 and 14 failures respectively, against an expectation of 20). Neither the $\chi^2$ nor the NIST failures indicate a cipher weakness; they are false rejections at the stated significance levels [26].

**S-box properties (RQ2, formal confirmation).** $\mathrm{DDT}_{\max} = 4$ and $\mathrm{LAT}_{\max} = 16$ across all 2,000 tested S-boxes, confirming the formal result from Section 6.1 over the full pool. The corresponding differential probability is $4/256 = 2^{-6}$ for every key.

**Serial independence.** The maximum absolute serial correlation at lags 1, 2, and 4 is 0.027, 0.021, and 0.024 respectively across the cold run — all negligible and indistinguishable from the values expected under a uniform random source.

**Timing and practical viability (RQ6).** The median per-encryption latency is 135,100 ns (cold) and 135,400 ns (warm), a difference of 300 ns — negligible and consistent with the absence of any meaningful warmup effect on the C-accelerated implementation. At a fixed plaintext length of 44 bytes, this corresponds to a throughput of approximately 0.31 MB/s on the test platform. The low coefficient of variation ($\mathrm{CV} < 5\%$ in 96.4% of cold runs and 94.7% of warm runs) confirms stable, predictable latency for the majority of keys. The $\approx 3.6\%$ of runs with $\mathrm{CV} \geq 5\%$ are attributable to OS scheduler interrupts rather than cipher behaviour, consistent with the right-tail-only trimming strategy in `story_basic.py`.

**Warmup effect.** The mean timing delta between the cold and warm runs is 166 ns (0.12% of the median latency), confirming that the C-accelerated implementation is not sensitive to CPU cache state at this message size. All cipher quality metrics (avalanche, SAC, BIC, key sensitivity, entropy, correlation) are stable across both runs to at least four significant figures, demonstrating that the test results are reproducible.