

# Enhancing the Hill Cipher with Byte-Level Substitution on the input Plaintext

By Tony Patti, May 6, 2026

This paper was inspired by a single sentence in Frank Rubin's book "Secret Key Cryptography". On page 259 he has this one sentence related to the Hill Cipher:

"You, the sender, can easily defeat such an attack by using a keyed simple substitution before and after the matrix multiplication."

This summary outlines our technical exploration into hardening a **Hill Cipher** using a large prime modulus and non-linear substitution on the input plaintext.

---

## 1. The Foundation: Hill Cipher mod(997727)

We began by discussing a high-dimensional Hill Cipher implementation utilizing the prime modulus  $p = 997727$ . While a standard Hill Cipher ( $C = M * P \text{ mod } p$ ) provides excellent **diffusion**—where one change in plaintext affects the entire ciphertext block—it is inherently linear. This linearity is its primary cryptographic weakness, as it allows an attacker to recover the secret matrix "M" using simple Gaussian elimination if they possess enough known plaintext/ciphertext pairs.

---

## 2. Introducing Non-Linearity: The SUB(P) Layer

To address the linearity weakness, we add a byte-level substitution layer *before* the matrix multiplication, the equation becomes:

$$C = M * \text{SUB}(P) \text{ mod } p$$

This makes the system significantly more resistant to:

- **Known-Plaintext Attacks (KPA):** The attacker can no longer use linear algebra to solve for M because the actual values being multiplied by the matrix are hidden behind the substitution.
  - **Chosen-Plaintext Attacks (CPA):** Attackers cannot use identity vectors to "read" the matrix columns directly, as the S-box obscures the input.
-

### 3. The Dual S-Box Architecture

Our prime modulus 997727 is large enough that we can utilize **two independent, secret S-boxes**—one for the **Left Byte** and one for the **Right Byte** of the input.

Since  $p=997727$  is a 20-bit prime, it can comfortably store the two bytes (16 bits) of the input. In this model, the input to the matrix is a composite of two non-linear transformations.

#### Advantages of the Dual-S-Box Model:

- **Asymmetry:** Using different S-boxes for the high and low bytes prevents symmetry-based cryptanalysis and ensures that identical byte values in different positions yield different inputs to the matrix.
- **Key Space Explosion:** By making the S-boxes part of the secret key, the total entropy of the system increases dramatically.
- **Avalanche Effect:** Small bit-changes in the input bytes result in unpredictable changes in the values processed by the matrix, fully utilizing the diffusion properties of the Hill Cipher.

---

### 4. Complexity and Entropy Analysis

Since an S-box is a permutation of 256 values, the number of possible S-boxes is the factorial  $256!$ , which is approximately  $8.5 * 10^{506}$ .

In terms of information theory, that number equates to **1,684 bits of entropy per S-box**.

For a system using two secret S-boxes, the theoretical key space added by the substitution layer alone is over **3,300 bits**. This dwarfs the 256-bit security level of modern standards like AES-256, making brute-force attacks on the substitution layer computationally impossible.

Focusing strictly on the **3,300 bits of additional entropy** provided by those two secret S-boxes, the benefit is not found in "mixing" (which the matrix already does), but in **Algebraic Obfuscation**. Here is the breakdown of what those 3,300 bits actually yield:

#### a. Breaking the "Unit Vector" Vulnerability

In a pure Hill Cipher, an attacker can perform a **Chosen-Plaintext Attack** by submitting a unit vector  $[1, 0, 0, \dots]$ . The resulting ciphertext is literally the first column of the secret matrix.

- **The Benefit:** With those 3,300 bits of S-box data, the attacker doesn't know what "1" becomes. It might become 847221 or 321. They can no longer "probe" the matrix to see its internal coefficients because they cannot control the actual values entering the multiplication.

## b. Elimination of the "Linearity Property"

A pure matrix cipher follows the rule:  $f(A + B) = f(A) + f(B)$ . This allows an attacker to use "differential" techniques to cancel out parts of the key.

- **The Benefit:** The 3,300 bits of secret substitution logic ensure that  $S(A+B)$  is NOT EQUAL TO  $S(A) + S(B)$ . This forces the cryptanalyst to treat the entire system as a **black box**. They can no longer use the relationship between two different plaintexts to derive information about the matrix.

## c. Protection Against "Known-Plaintext" Matrix Inversion

If an attacker has "n" plaintext/ciphertext pairs for a standard Hill Cipher, they solve a system of linear equations:  $M = C * P^{-1}$ .

- **The Benefit:** Because the S-boxes are part of the key (and thus unknown), the attacker does not have the values for P. They only have the values for the *pre-substituted* bytes. To run the inversion, they would first have to guess the 1,684 bits of the first S-box and the 1,684 bits of the second. This effectively moves the attack from "polynomial time" (easy) to "exponential time" (impossible).

## d. Resistance to Statistical Analysis: The "Relabeling Shield"

If the input data has a biased distribution (e.g., English text where "e" is frequent), a pure Hill Cipher can be vulnerable to frequency analysis. The secret S-boxes act as a Relabeling Shield; while a 1:1 mapping does not change the frequency distribution itself, it divorces the "peaks" in the data from their numerical identity. This defense is particularly effective against common two-byte patterns like "th," "he," or "in."

In this architecture, a two-byte sequence is processed with independent left-and-right byte substitutions. Even if "th" is statistically the most frequent pair in your plaintext, the attacker cannot utilize that frequency to crack the matrix because they do not know the secret values those bytes were mapped to. By re-mapping these common values to arbitrary, secret points in the  $\text{mod}(99727)$  field, you ensure that while a pattern may exist, its mathematical meaning is hidden. Because Hill Cipher cryptanalysis requires exact numerical inputs to solve the linear equations for the secret matrix M, hiding these identities behind 3,300 bits of secret mapping makes the linear algebra impossible to resolve.

---

## 5. L1 Cache: The Nanosecond Lookup

A one-byte substitution via an array lookup nominally takes 1 nanosecond. Below are the three factors which influence the timing.

**L1 Cache Hit:** Because a 256-byte array is so small, and repeatedly utilized for every byte of the input plaintext, it will remain resident in the L1 cache. On a modern Intel or AMD processor, an L1 cache hit typically takes about 4 to 5 clock cycles. On a 3.0 GHz processor, 3 cycles equals exactly 1 nanosecond.

**Pipelining and Superscalar Execution:** Modern CPUs execute multiple instructions per cycle. When performing this substitution in a loop (e.g., across an entire plaintext block), the throughput improves such that the "amortized" cost can drop below 1 cycle per byte (effectively < 0.3 ns) due to the CPU's ability to pre-fetch data and execute the lookup in parallel with other arithmetic.

**Branch Prediction:** Since a substitution lookup is a simple indexed memory access `SUB[input_byte]`, there is no branching involved. This makes the timing highly deterministic, providing a robust defense against timing-based side-channel attacks.

---

## 6. Summary of Benefits

While the matrix provides the **width** (diffusion), the 3,300 bits of secret substitution provide the **depth** (complexity).

Without the S-boxes, the cipher is a high-speed "shredder" that can be put back together with basic linear algebra. With the S-boxes, we have effectively locked the shredder inside a vault for which the attacker doesn't have the combination. We have turned a problem of **solving equations** into a problem of **searching an impossibly large space**.

The hybrid approach of combining **Secret Dual S-Boxes** with a **Large-Prime Hill Cipher** creates a robust cryptographic primitive. It balances the high-speed linear diffusion of matrix multiplication with the high-entropy non-linearity of key-dependent substitution. This design effectively forces an analyst away from simple algebraic attacks and toward much more difficult differential or linear cryptanalysis.

---

## References

1. Rubin, Frank (2022) book "Secret Key Cryptography".
2. Hill, Lester S. (1929). "Cryptography in an Algebraic Alphabet". The American Mathematical Monthly.
3. Hill, Lester S. (1931) "Concerning Certain Linear Transformation Apparatus of Cryptography". The American Mathematical Monthly, March 1931, pages 135 - 154.
4. Cooper, Rodney H. (1980). "Linear Transformations in Galois Fields and their Application to Cryptography". Cryptologia
5. Dooley, John (January 1, 2018). "10.1 The Shoulders of Giants: Friedman, Hill, and Shannon". History of Cryptography and Cryptanalysis: Codes, Ciphers, and Their Algorithms. Springer. p. 167. ISBN 978-3-319-90442-9
6. Wikipedia "Substitution–permutation network".  
[https://en.wikipedia.org/wiki/Substitution–permutation\\_network](https://en.wikipedia.org/wiki/Substitution–permutation_network)
7. Wikipedia "Lester S. Hill" [https://en.wikipedia.org/wiki/Lester\\_S.\\_Hill](https://en.wikipedia.org/wiki/Lester_S._Hill)
8. Patti, Tony (2025) "An Interesting Example at the Intersection of Matrix Mathematics and Cryptography - Version 2". <https://cryptosystemsjournal.com/an-interesting-example-atthe-intersection-of-matrix-mathematics-and-cryptography-version-2.pdf>
9. Patti, Tony (2026) "Extending the Hill Cipher to be Asymptotic to a One-Time Pad" <https://cryptosystemsjournal.com/Extending-the-Hill-Cipher-to-be-Asymptotic-to-a-One-Time-Pad.pdf>
10. Patti, Tony "Introducing the Hill-GF-4D Cryptosystem! Four-dimensional Matrix-within-Matrix Cryptosystem inspired by the last paragraph of Lester Hill's 1931 paper! <https://cryptosystemsjournal.com/matrix-within-matrix.html>
11. Shannon, Claude. (1948). "A Mathematical Theory of Communication". The Bell System Technical Journal, Volume 27, pages 379–423, 623–656, July, October, 1948.  
<https://people.math.harvard.edu/~ctm/home/text/others/shannon/entropy/entropy.pdf>
12. Shannon, Claude. (1949). "Communication Theory of Secrecy Systems". Bell System Technical Journal. Volume 28-4, pages 656-715, October 1949).  
<https://pages.cs.wisc.edu/~rist/642-spring-2014/shannon-secrecy.pdf>