Lightweight Cryptography - Toy Cipher Exploration

Kriti Arora

November 2024

---

1 Overview of the Toy Cipher

The original Saturnin cipher operates on a 16×16 state, i.e., a matrix of 16 rows and 16 columns of 4-bit cells, totaling 256 bits. In contrast, our toy variant significantly reduces the state size for analysis and experimentation.

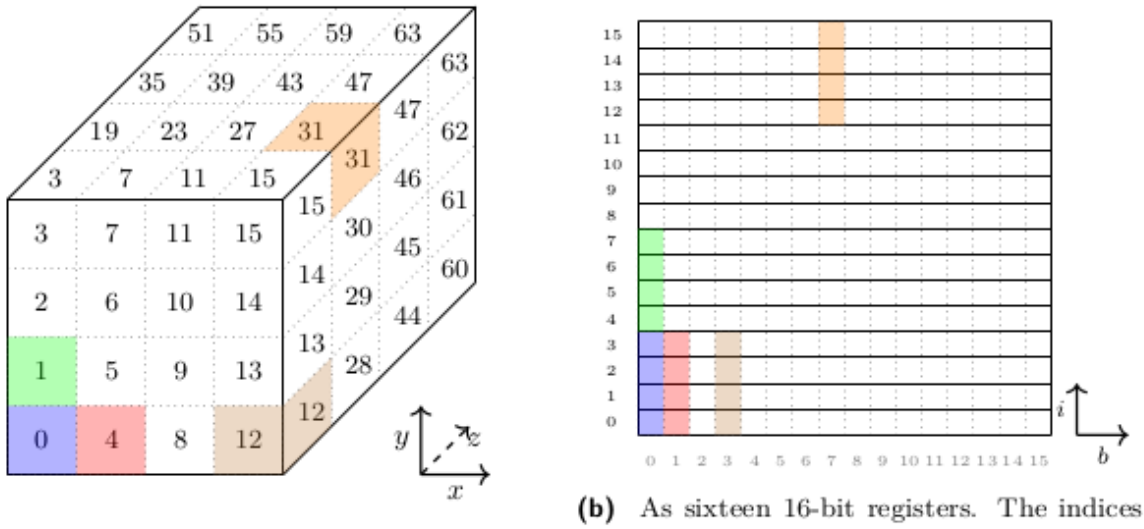
Toy Cipher State. The toy design uses an 8×4 state, consisting of:

$$8 \text{ registers} \quad \times \quad 4 \text{ bits each} \; = \; 32 \text{ bits}.$$

These registers are denoted

$$(x_0, x_1, x_2, x_3,\ x_4, x_5, x_6, x_7),$$

where each x_i is a 4-bit value.



(a) As a $4 \times 4 \times 4$ cube of 4-bit nibbles. The boundaries between the nibbles are in gray.

(b) As sixteen 16-bit registers. The indices and boundaries of the registers are in black, those of the bits are in gray.

Figure 1: The two representations of the 256-bit state of SATURNIN. Nibbles and their corresponding bits are represented with the same color in each representation.

Figure 1: **Full Saturnin state (16×16).**

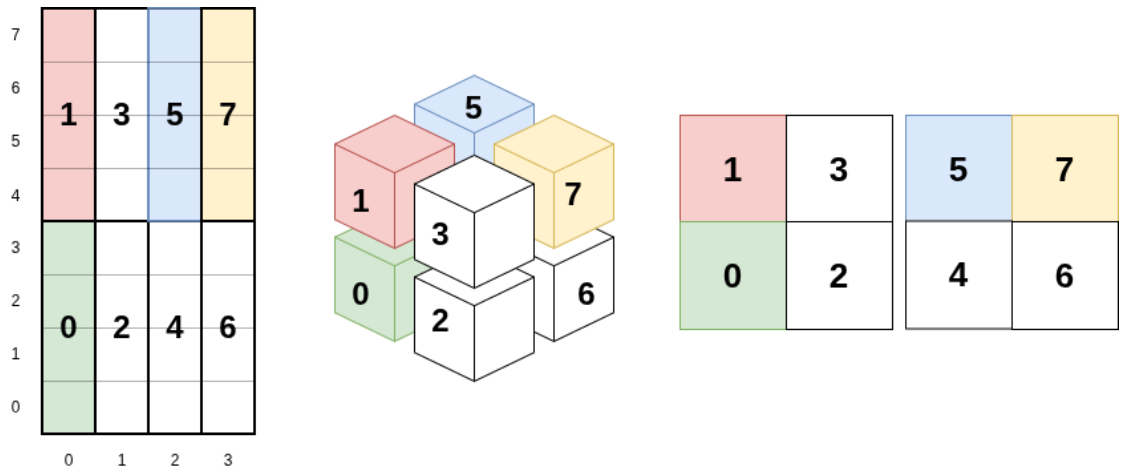


Figure 2: **Toy cipher state (8 registers of 4 bits each).**

2 Nonlinear Layer: Bitsliced S-box Application

The cipher state consists of eight 4-bit registers:

$$\mathbf{S} = (x_0, x_1, x_2, x_3,\ x_4, x_5, x_6, x_7), \qquad x_i \in \{0, \ldots, 15\}.$$

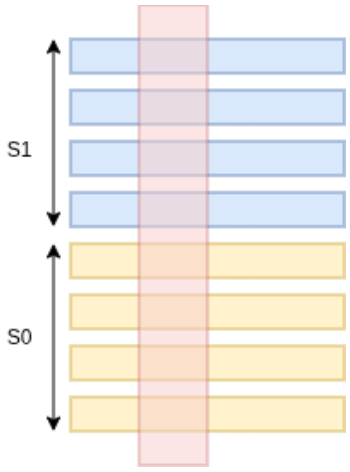


Figure 3: **Bitsliced S-box structure**

2.1 Bitslice Structure

To apply the nonlinear layer, we use a *bitsliced* implementation of the S-box. The eight registers are divided into two groups:

$$\mathcal{G}_0 = (x_0, x_1, x_2, x_3), \qquad \mathcal{G}_1 = (x_4, x_5, x_6, x_7).$$

Each group contains four 4-bit elements. For a group

$$\mathcal{G} = (u_0, u_1, u_2, u_3),$$

the bitsliced representation is constructed by collecting the same bit position from each element. Thus, for bit position $j \in \{0, 1, 2, 3\}$, we define the slice

$$(a_j, b_j, c_j, d_j) = \big((u_0)_j,\ (u_1)_j,\ (u_2)_j,\ (u_3)_j\big),$$

where $(u_i)_j$ denotes the j-th bit of u_i.

Each quadruple (a_j, b_j, c_j, d_j) forms one S-box input. Since each group has four bit positions and the cipher has two groups, the design applies

$$\boxed{\text{8 S-boxes in parallel}}$$

— four from $\mathcal{G}_0$ and four from $\mathcal{G}_1$.

2.2 Group-specific S-box Variants

The two groups use different S-box variants:

- S_0 is applied to all four slices of the first group $\mathcal{G}_0$.
- S_1 is applied to all four slices of the second group $\mathcal{G}_1$.

Thus,

$$\mathcal{G}_0 \xrightarrow{\text{bitslicing}} (a_j, b_j, c_j, d_j) \xrightarrow{S_0} (a'_j, b'_j, c'_j, d'_j) \xrightarrow{\text{recombine}} \mathcal{G}'_0,$$

and similarly for $\mathcal{G}_1$ with S_1.

3 Shift Operations: SR_slice and SR_sheet

After the nonlinear layer, the cipher applies two shift transformations: SR_slice and SR_sheet. Both operate on different structural views of the state and contribute to inter-register diffusion.

3.1 SR_slice

In the sliced representation of the state, each slice consists of four 4-bit registers arranged conceptually as a small 2×2 grid.

The SR_slice transformation performs a horizontal rotation on the *top row* of each slice, while the bottom row remains unchanged:

bottom row: unchanged, top row: shifted left by 1 position.

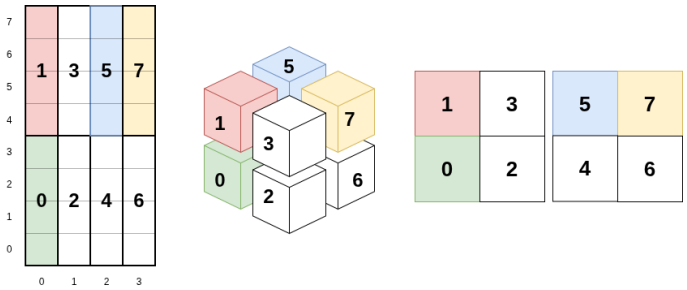


Figure 4: **SR_slice: State before applying the slice shift.**

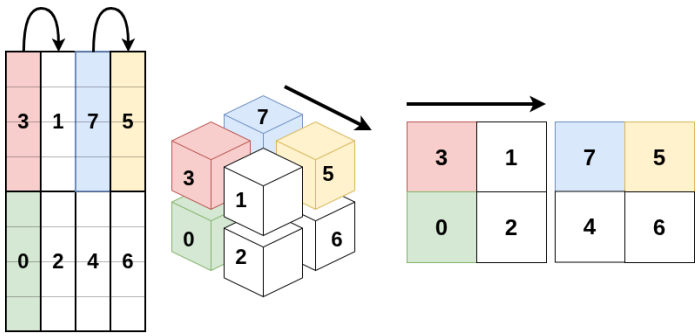


Figure 5: **SR_slice: Only the top row is shifted; bottom row unchanged.**

3.2 SR_sheet

In the sheet-based view of the state, each sheet again forms a 2×2 substructure of the 8×4 state. The SR_sheet transformation mirrors the behavior of SR_slice, but now at the sheet level.

bottom row: unchanged, top row: shifted left by 1 position.

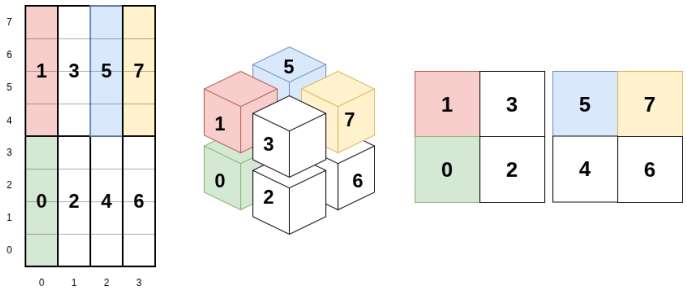


Figure 6: **SR_sheet: Sheet before shifting.**

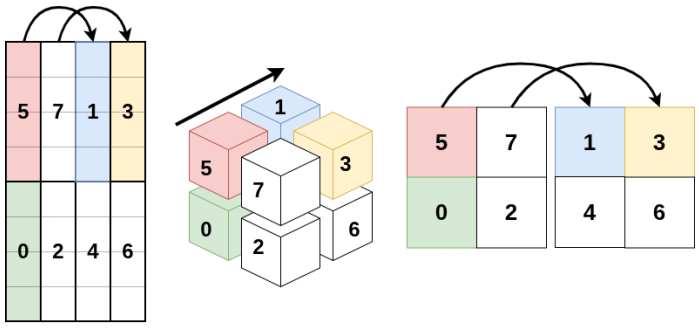


Figure 7: **SR_sheet: Top row shifted; bottom row unchanged.**

4 MDS Layer

The Mix-Diffusion Step (MDS) provides linear diffusion across the state. In the original Saturnin design, the 16 registers of each column are divided into four groups of four registers, which are denoted

$$(a, b, c, d).$$

4.1 Original Saturnin MDS

Each group (a, b, c, d) undergoes a local rotation-and-mix operation. Let the four registers in the group be written as

$$(t_0, t_1, t_2, t_3).$$

Saturnin applies the following transformation:

$$(t_0, t_1, t_2, t_3) \longleftarrow (t_1,\ t_2,\ t_3,\ t_0 \oplus t_1).$$

This operation is repeated for all four groups in the column. Because every output depends on multiple inputs, the entire column becomes strongly diffused.

4.2 Toy Cipher MDS

In our reduced-state variant, the column contains only 8 registers. Therefore, we divide them into four groups of two registers each:

$$(a, b, c, d),$$

where each letter represents a pair of 4-bit registers.

Let one such pair be written as

$$(t_0, t_1).$$

The toy cipher applies a simplified MDS mixing step:

$$(t_0, t_1) \longleftarrow (t_1,\ t_0 \oplus t_1).$$

Thus, each pair undergoes a rotation combined with XOR mixing. The remaining structural arrangement of the MDS follows the original Saturnin design principles, but adapted to the smaller 8×4 state.

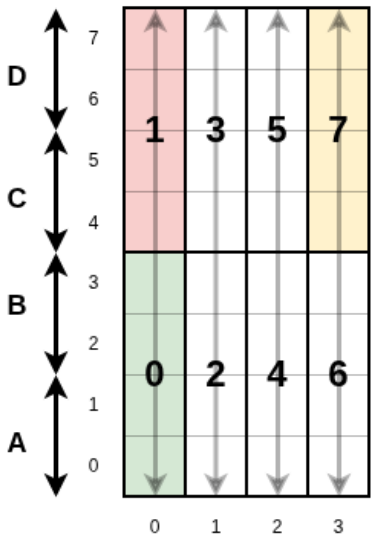


Figure 8: **Toy cipher MDS**

5 Empirical Properties of the MDS Layer

To evaluate the correctness and diffusion quality of the proposed MDS transformation, we performed a series of tests on the 8-register, 4-bit-per-register state used in the toy cipher.

5.1 Correctness of the MDS and Its Inverse

A correctness check was conducted to verify that the inverse MDS operation precisely reverses the forward transformation. For a sample input state

$$(0, 0, 0, 1,\ 0, 0, 0, 0),$$

the forward MDS produced

$$(0, 1, 1, 0,\ 0, 1, 0, 0),$$

and applying the inverse MDS returned

$$(0, 0, 0, 1,\ 0, 0, 0, 0).$$

Thus the correctness condition

$$\text{InvMDS}(\text{MDS}(x)) = x$$

holds for this and all tested inputs.

5.2 Diffusion Measurement

To quantify diffusion, each input bit was flipped individually and the number of output nibbles that changed was measured and averaged.

The measured average diffusion was:

$$\boxed{\text{3.38 output nibbles changed per input bit flip}}.$$

This indicates moderate diffusion: each single-bit change affects, on average, more than three of the eight output nibbles, demonstrating that the linear layer contributes meaningfully to spreading differences across the state.
