Implementation and Analysis of Mysterion

Meghana Varanasi[1], Shreya Gundu[2] and Debajyoti Halder[3]

[1] IIT Bhilai, Raipur, India, meghana@iitbhilai.ac.in
[2] IIT Bhilai, Raipur, India, gundus@iitbhilai.ac.in
[3] IIT Bhilai, Raipur, India, debajyotih@iitbhilai.ac.in

Abstract. Mysterion is an instance of XLS-design which is a family of cipher with efficient bitslice implementation against side channel attack. We discuss Mysterion and its security against various attacks including invariant subspace attack and boomerang attack which is due to the XLS-design, a massive upgrade from LS-designs. We discuss the faults in LS-design and how combining Super S-boxes with ShiftColumns prevented attacks and gave sufficient security margin to Mysterion through XLS-design. A simple block cipher can still be very much secure against physical attacks.

Keywords: Mysterion, XLS-design, Invariant Subspace Attack

Contents

1 Introduction

LS-designs presented us with efficient bit slicing. They were combinations of linear diffusion L-boxes and non-linear bitslice S-boxes. However LS-designs were susceptible to invariant subspace attack which led to doubts on the proposed instances of LS-designs. XLS-designs prevented the attacks with heuristic changes. A better choice of round constants prevented the propagation of invariant subspace for the S-boxes or L-boxes. We now have an eXtensible LS-design.

In this paper we demonstrate Mysterion, an instance of XLS-design comprising of Super S-boxes, made of optimal components - 4-bit S-boxes and 32-bit L-boxes based on Maximum Distance Separable (MDS) code, and ShiftColumns operation. We show security margins of Mysterion and it's similarity with AES-like cipher because of the combined S-boxes and ShiftColumns design.

2 Mathematics

1. **Galios Field :** Galois field is a field that contains a finite number of elements. As with any field, a finite field is a set on which the operations of multiplication, addition, subtraction and division are defined and satisfy certain basic rules. The most common examples of finite fields are given by the integers mod p when p is a prime number.

2. **Companion matrix :** In linear algebra, the Frobenius companion matrix of the monic polynomial $p(t) = c_0 + c_1 t + \cdots + c_{n-1}t^{n-1} + t^n$, is the square matrix defined as

$$C(p) = \begin{bmatrix} 0 & 0 & \dots & 0 & -c_0 \\ 1 & 0 & \dots & 0 & -c_1 \\ 0 & 1 & \dots & 0 & -c_2 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \dots & 1 & -c_{n-1} \end{bmatrix}$$

3. **MDS matrix :** An MDS matrix (Maximum Distance Separable) is a matrix representing a function with certain diffusion properties that have useful applications in cryptography. Technically, an m×n matrix A over a finite field K is an MDS matrix if it is the transformation matrix of a linear transformation $f(x) = Ax$ from Kn to Km such that no two different $(m + n)$-tuples of the form $(x, f(x))$ coincide in n or more components. Equivalently, the set of all $(m + n)$-tuples $(x, f(x))$ is an MDS code, i.e. a linear code that reaches the Singleton bound.

4. **BCH code :** In coding theory, the BCH codes or Bose–Chaudhuri–Hocquenghem codes form a class of cyclic error-correcting codes that are constructed using polynomials over a finite field (also called Galois field).

3 xLS Design

3.1 Bit Slicing

There are 2 basic ideas behind bit slicing :

1. converting the cipher into bit-wise operations (like the way we'd implement it in hardware)

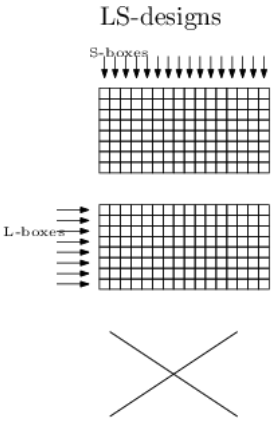
2. carrying out those bit wise operations in parallel (We show two implementations of the sbox, one just showing the bitwise operation on each 4 bit input as well as on a block (8x4 i.e.,applying it paralelly on 8 inputs of size 4 bits))

Bitwise slicing is efficent in the following cases :

1. Bit shifts, rotations and other bit permutations
2. Bitwise logical operations
3. Addition and subtraction
4. multiplication and division
5. Table lookups
6. Conditional code

3.2 LS and xLS Designs

3.2.1 Brief on LS Design

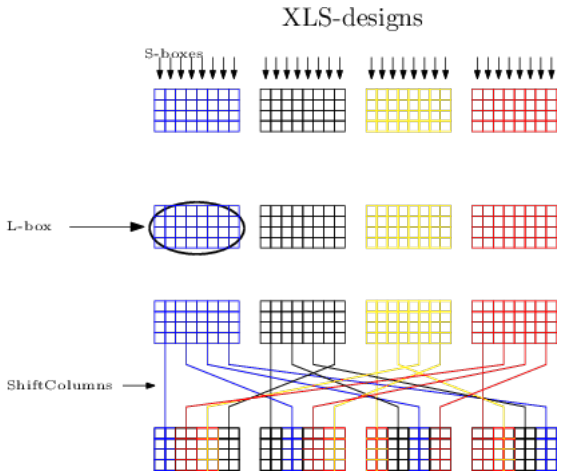


LS-designs, proposed at FSE 2014, aimed for efficient bitslice implementation. LS-designs, as it is obvious by the name, are combinations of linear diffusion L-boxes and non-liner bitslice S-boxes.

Among the two proposed instances - Robin (involutive cipher) and Fantomas (non-involutive cipher), involutive instance Robin is susceptible to an invariant subspace attack, and has lead to a weak key set of density 2^{-32}.

The invariant subspace attack against Robin can be prevented with simple heuristics i.e. better choice of round constants. In order to avoid invariant subspaces for the S-boxes or L-boxes to be propagated through the rounds, the round constants should have varying bits (in bitslice representation).

3.2.2 XLS Design - the difference



eXtensible LS (XLS) Design is based on 32-bit "Super S-Boxes" (4-bit S-boxes and 32-bit L-boxes based on a Maximum Distance Separable (MDS) code), which are further combined with additional ShiftColumns operation to obtain 128 and 256 bit ciphers.

AES also had a similar approach, using S-boxes and then combining with ShiftRows (also Mix Columns and round keys). However here bitslice is used rather than a block structure.

Few differences pointers:

1. its diffusion layer is not based on binary matrices anymore (non-binary MDS code is used with S-boxes with smaller bit sizes)
2. it requires an additional ShiftColumns operations

4 Analysis

Algorithm : XLS-design with $l \cdot s$ -bit L-boxes, s -bit S-boxes and b blocks

$x \leftarrow P \oplus K$ ▷ x is a $s.(l.b)$ bits matrix
for $0 \leq r < N_r$ **do**
  for $0 \leq j < b$ **do**
    for $0 \leq i < l$ **do**
      $x[j, \star, i] = \mathrm{S}[x[j, \star, i]]$; ▷ **S**-box layer
    for $0 \leq j < b$ **do**
      $x[j, \star, \star] = \mathrm{L}[x[j, \star, \star]]$ ▷ **L**-box layer
    for $0 \leq k < s$ **do**
      $x[\star, k, \star] =$ Shift Columns $[x[\star, k, \star]]$ ▷ **ShiftColumns** layer
  $x \leftarrow x \oplus K \oplus C(r)$ ▷ Key and round constant addition return
return x

4.1 Sbox

Mysterion uses S-boxes that has bitslice representation with a combination of AND (also OR) and XOR gates. The class-13 S-box, as used by Mysterion, has a bitslice representation of 4 AND (precisely 3 AND and 1 OR) gates and 4 XOR gates. It has an algebraic degree of three and as shown below - differential probability 2^{-2} and linear probability 2^{-1}

Algorithm :

Algorithm 1: S-box, bitslice representation

Require: 4 input bits (A,B,C,D)
Ensure: 4 output bits such as (a,b,c,d)= S(A,B,C,D)
1: $a = A$ & B;
2: $a = a \bigoplus C$;
3: $c = B \mid C$;
4: $c = c \bigoplus D$;
5: $d = a$ & D;
6: $d = d \bigoplus A$;
7: $b = c$ & A;
8: $b = b \bigoplus B$;
9: **return** (a, b, c, d)

4.1.1 DDT

As you can see from the table, the Differential probability = $\frac{4}{16} = 2^{-2}$

Table 1: DDT of Sbox

	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**a**	**b**	**c**	**d**	**e**	**f**
1	4	.	.	.	4	.	.	4	.	.	.	.	4	.	.
2	.	.	.	4	.	4	.	.	.	2	2	2	2	.	.
3	2	.	2	.	2	.	2	.	2	.	2	2	.	2	.
4	.	.	.	.	.	.	.	4	4	2	2	.	.	2	2
5	4	.	.	.	4	.	.	.	.	2	2	.	.	2	2
6	.	4	.	4	.	.	.	.	.	.	.	2	2	2	2
7	2	.	2	.	2	.	2	.	2	2	.	2	.	.	2
8	.	4	4	.	.	4	4	.	.	.	.	.	.	.	.
9	.	2	2	.	.	2	2	.	.	2	2	.	.	2	2
a	.	.	.	4	.	.	4	.	.	2	2	2	2	.	.
b	2	2	.	.	2	2	.	.	2	2	.	2	.	.	2
c	.	.	.	.	.	.	.	4	4	2	2	.	.	2	2
d	.	2	2	.	.	2	2	4	.	.	.	.	4	.	.
e	.	.	4	4	.	.	.	.	.	.	.	2	2	2	2
f	2	2	.	.	2	2	.	.	2	.	2	2	.	2	.

4.1.2 LAT

As you can see from the table, the Linear probability = $\frac{4}{8} = 2^{-1}$

Table 2: LAT of Sbox

	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**a**	**b**	**c**	**d**	**e**	**f**
1	4	.	-4	.	.	.	.	.	4	.	4	.	.	.	.
2	.	.	.	4	4	4	-4	.	.	.	.	.	.	.	.
3	.	.	.	4	.	-4	.	.	.	.	.	.	4	.	4
4	2	.	2	.	-2	.	-2	4	2	.	-2	.	2	4	-2
5	2	.	2	.	2	.	2	4	2	.	-2	.	-2	-4	2
6	-2	.	-2	.	2	.	2	4	-2	.	2	4	2	.	-2
7	2	.	2	.	2	.	2	-4	2	.	-2	4	2	.	-2
8	.	-4	.	2	-2	2	2	.	.	4	.	2	-2	2	2
9	4	.	.	-2	2	2	2	.	-4	.	.	-2	2	2	2
a	.	4	.	-2	2	-2	-2	.	.	4	.	2	-2	2	2
b	.	.	4	2	2	-2	2	.	.	.	4	-2	-2	2	-2
c	-2	4	-2	2	.	2	4	.	2	.	-2	-2	.	2	.
d	-2	.	2	-2	.	2	.	.	2	-4	2	2	.	2	4
e	2	4	2	2	-4	2	.	.	-2	.	2	2	.	-2	.
f	-2	.	2	-2	.	2	.	.	2	4	2	-2	4	-2	.

4.2 Lbox

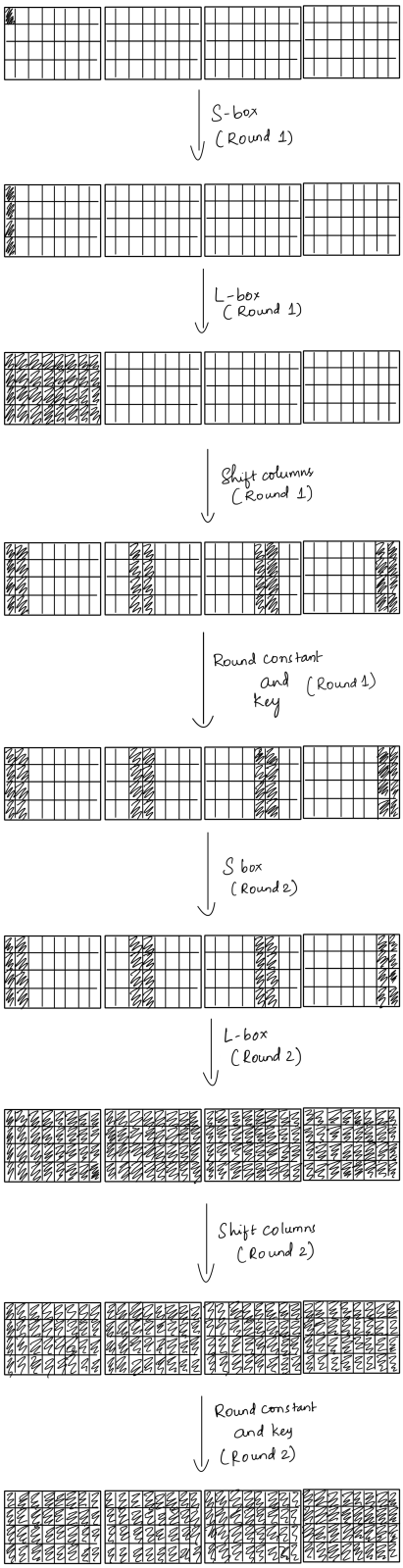
$$C = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & \alpha^3 & \alpha^4 & \alpha^{12} & \alpha^8 & \alpha^{12} & \alpha^4 & \alpha^3 \end{pmatrix}, C^8 = \begin{pmatrix} 1 & \alpha^3 & \alpha^4 & \alpha^{12} & \alpha^8 & \alpha^{12} & \alpha^4 & \alpha^3 \\ \alpha^3 & \alpha^{13} & \alpha^4 & \alpha & 1 & \alpha^2 & \alpha^2 & \alpha^{12} \\ \alpha^{12} & \alpha^{14} & \alpha^{12} & \alpha^{14} & \alpha^2 & \alpha^7 & \alpha^5 & \alpha^8 \\ \alpha^8 & 1 & \alpha^5 & \alpha^{14} & \alpha^7 & \alpha & \alpha^2 & \alpha^3 \\ \alpha^3 & \alpha^{14} & \alpha^9 & \alpha^{10} & \alpha^{10} & \alpha^9 & \alpha^{14} & \alpha^3 \\ \alpha^3 & \alpha^2 & \alpha & \alpha^7 & \alpha^{14} & \alpha^5 & 1 & \alpha^8 \\ \alpha^8 & \alpha^5 & \alpha^7 & \alpha^2 & \alpha^{14} & \alpha^{12} & \alpha^{14} & \alpha^{12} \\ \alpha^{12} & \alpha^2 & \alpha^2 & 1 & \alpha & \alpha^4 & \alpha^{13} & \alpha^3 \end{pmatrix}.$$

C is the companion matrix of the polynomial
C^8 is the underlying matrix of the Mysterion l-box

Mysterion Lbox uses linear transformation. The paper by Augot and Finiasz[9] proposes an algorithm which finds recursive MDS diffusion layers using shortened BCH codes. What this algorithm does is it takes degree of the polynomial k (hence the size of the companion matrices), and the field size $q = 2^s$ as parameters, and provides all the polynomials of degree k over F_{2^s} such as their companion matrices raised to the power k gives MDS diffusion layers.

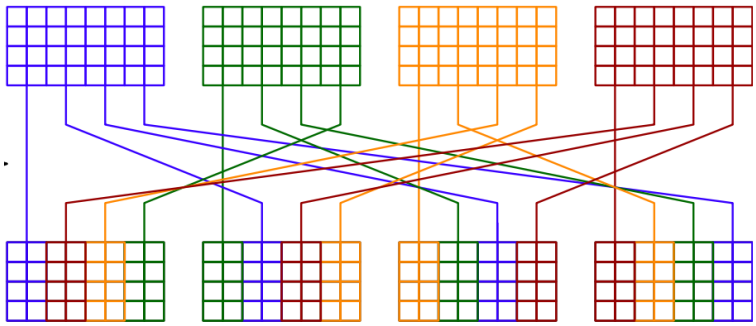
Selected polynomial, obtained after this is run using Magma code with parameters k = 8 and s = 4, has its coefficients in $\mathbb{F}_{2^4}$ is $\mathrm{P(X)} = \mathrm{X}^8 + \alpha^3 \cdot \mathrm{X}^7 + \alpha^4 \cdot \mathrm{X}^6 + \alpha^{12} \cdot \mathrm{X}^5 + \alpha^8 \cdot \mathrm{X}^4 + \alpha^{12} \cdot \mathrm{X}^3 + \alpha^4 \cdot \mathrm{X}^2 + \alpha^3 \cdot \mathrm{X} + 1$. Its differential and linear branch number equal to 9.

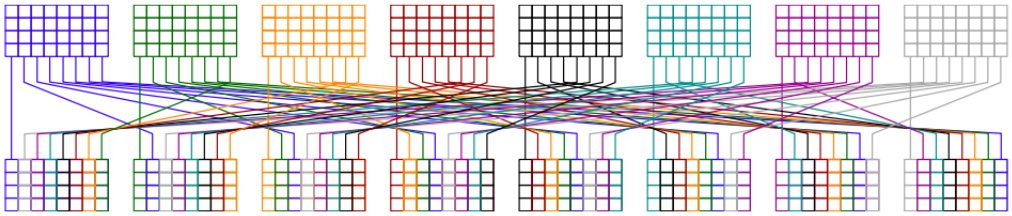
Its purpose is to diffuse changes in the state. Because of this lbox, input change in one bit is diffused to entire state is just 2 rounds. LBox is implemented by multiplying the state with MDS matrix. The multiplying matrix was its companion matrix to the power eight therefore, we could replace one matrix multiplication by eight vector multiplications (This is what is done in the implementation).


S-box
(Round 1)
L-box
(Round 1)
Shift columns
(Round 1)
Round constant
and
Key
(Round 1)
S box
(Round 2)
L-box
(Round 2)
Shift columns
(Round 2)
Round constant
and key
(Round 2)

4.3 Shift Columns

In Mysterion-128, as there are 4 blocks (with each block having 8 columns) ShiftColumns acts on columns two by two. The first two columns of each block will be at the same position , the second two columns are moved by one block, the third two columns are moved by two blocks, and the fourth two columns are moved by three blocks. But when coming to the Mysterion-256 as there are 8 blocks (with each block having 8 columns) ShiftColumns acts on columns one by one. The first column of each block will be at the same position. The nth column of the block will be moved by n-1 blocks (where n= 2 to 8)





4.4 Key and constant addition

Up until now, we have done nothing to make the ciphertext dependent on the key. So to do this, we add the key to the state. Because we want every round to be different, we also add a round-specific constant value.
The Mysterion block cipher has no key schedule. In every round, the same key is added to the state.

5 Security Analysis

5.1 Boomerang Attack

This is a special type of Differential Cryptanalysis. The basic idea of this attack is to find two differentials for the two sub ciphers C_0 and C_1 obtained from cipher C such that $C = C_0 \circ C_1$ instead of finding one single differential for C. Since shorter differentials have more better probabilities, this might improve the results for the attacker. Now to find the best differential probability of the a boomerang distinguisher, lets use 2 pre established theorems from paper [9] :
Theorem : Four rounds of Mysterion-128 has at least 45 active S-boxes.
Theorem : Four rounds of Mysterion-256 has at least 81 active S-boxes.
Therefore we have :

For 128 bit :

$$\Pr_{diff}(4R) \leq \Pr_{diff}^{max}(Sbox)^{45} = (2^{-2})^{45} = 2^{-90}$$

$$\implies \Pr_{diff}^{max}(8R) = \Pr_{diff}(4R) * \Pr_{diff}(4R) = 2^{-90} * 2^{-90} = 2^{-180}$$

the Pr obtained is way less that brute force probability (2^{128}).
Therefore, 8 round 128 bit Mysterion is secure against Boomerang Attack.

Similarly, now coming to 256 bit Mysterion :
We use the theorem 2 above to find the max differential prob that can be obtained :

$$\Pr_{diff}^{max}(8R) = \Pr_{diff}(4R) * \Pr_{diff}(4R) = 2^{-81*2} * 2^{-81*2} = 2^{-324} >> 2^{-256}$$

the Pr obtained is way less that brute force probability (2^{256}).
Therefore, 8 round 256 bit Mysterion is secure against Boomerang Attack.

Therefore Mysterion Cipher is secure against Boomerang Attack.

5.2 Integral Attack

Considering a collection of chosen plaintexts of m-bytes and their corresponding ciphertexts, integral attacks tries to extract information about the key by observing the sum of ciphertext values. Block ciphers based on SPNs can be tried with this attack with great efficiency. For AES, we may find integral property upto 4 rounds, and then we can mount this attack from 7-9 rounds depending on the key size. But for Mysterion after we find integral property for 4 rounds, we remain with sufficient security margin for the full cipher, since there are 12 and 16 rounds for Mysterion -128 and Mysterion-256.

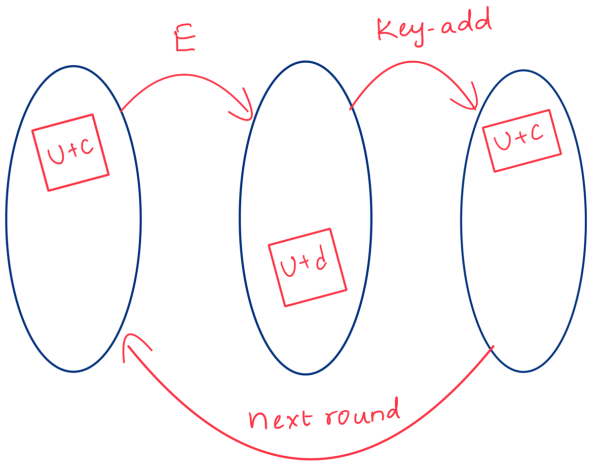
In EUROCRYPT 2015 division property was introduced. It allows to construct more efficient integral distinguishers exploiting the limited algebraic degree of reduced ciphers. However Mysterion still has sufficient security margins for this attack also.

5.3 Invariant subspace Attack

General Abstraction:
Let us consider an n-bit iterative block cipher, with round function R_k where,
$R_k : \mathbb{F}_2^n \text{ X } \mathbb{F}_2^n \to \mathbb{F}_2^n$, such that $R_k(x) = E(x + k)$, with E as an n-bit permutation.
If there exists a subspace $U \subseteq \mathbb{F}_2^n$ and two constants $c, d \in \mathbb{F}_2^n$ such that $E(U + c) = U + d$, then for a round key $k = u + c + d$ with $u \in U$, the following holds:

$$R_k(U + d) = E((U + d) + (u + c + d)) = E(u + c) = U + d.$$

i.e. the round function maps the affine subspace $U + d$ onto itself. If all round keys are in $k \in U + (c + d)$, then this property is iterative over arbitrary number of rounds.



LS Design models are vulnerable to this attack . This was identified by performing an exhaustive analysis on a 32 bit block. By adding the *Shift columns* operation to LS design models (which are called as XLS models) ,this attack can no longer be done . Because, the shift columns operation prevents the propagation of subspace found for the L-box with high probability.

Hence , Mysterion being an XLS design model is resistant to invariant subspace attacks even with sparse round constants.

6 Conclusion

Mysterion is an example that block ciphers can be simple yet powerful and secure. XLS-design based models of ciphers have sufficient security margins for physical attacks. However grasping L-box and a few more concepts posed as a challenge to us. Implementations show that the Mysterion is very fast and is apt for lightweight uses. Although Mysterion-256 is still a bit misty to us we have been successful in implementing a working Mysterion-128 cipher.

7 Links

1. Implementation Source Code Link
2. Google Collab Link
3. Online Tool Link
4. Online Tool Source code

References

[1] https://dsprenkels.com/mysterion.html

[2] https://link.springer.com/chapter/10.1007/978-3-662-46706-0_2

[3] https://www.researchgate.net/profile/Francois-Xavier_Standaert/publication/297725067_Improving_the_security_and_efficiency_of_

block_ciphers_based_on_LS-designs/links/5c50199da6fdccd6b5d1a888/Improving-the-security-and-efficiency-of-block-ciphers-based-on-LS-designs.pdf

[4] https://en.wikipedia.org/wiki/Finite_field

[5] https://en.wikipedia.org/wiki/Companion_matrix

[6] https://en.wikipedia.org/wiki/MDS_matrix

[7] https://en.wikipedia.org/wiki/BCH_code

[8] https://perso.uclouvain.be/fstandae/PUBLIS/157b.pdf

[9] https://perso.uclouvain.be/fstandae/PUBLIS/157.pdf

[10] https://www.cryptolux.org/index.php/Lightweight_Block_Ciphers