Inside Rijndael

Understanding of the arithmetic in Rijndael's finite field

Fernando Crema Garcia

Data Privacy and Security
Laurea Magistrale in Data Science
Sapienza, University of Rome

A. Y. 2019 - 2020

Table of contents

Outline

History

1. In 1997, NIST decided to deprecate DES and focus on DES3 while opening proposals for the new Advanced Encryption Standard (AES) (FIPS PUB 197).
2. Several characteristics were mandatory for the candidate [PP10]:
   - Block cipher with 128 bit block size
   - Three key lengths must be supported: 128, 192 and 256 bit.
   - Security relative to other submitted algorithms.
   - Efficiency in software and hardware
3. Candidates narrowed down to 5 in August of 1999.
4. In October of 2000 NIST should announce the winner.
5. In November of 2001, AES was formally approved as a **US federal standard**.

The winner

In the end, the elegance, efficiency, security, and principled design of Rijndael won the day for its two Belgian designers, Joan Daemen and Vincent Rijmen.

About this project

Objective: Gain a better understanding of the arithmetic in Rijndael's finite field (underlying the AES block-cipher).

Output: A program able to:

- Perform the basic operations in Rijndael's finite field.
  - Sum (and subtraction).
  - Multiplication.
  - Division.
- Use the extended Euclidean algorithm [AJM96] for computing the multiplicative inverse of a polynomial.
- Use your program to re-generate the AES S-Boxes (for both encryption and decryption).

What is not

Even though the complete implementation of the AES is really interesting we will focus only on the under understanding of the arithmetic in Rijndael's finite field and the associated operations and implementations.

Bib load

- [AJM96]
- [Ven12]
- [KL14]
- [JD20]

Outline

Group

An group [PP10] $(G, +)$ consists of a set G and an operation defined on its elements, here denoted by $\circ$:

$$\circ : G \times G \to G : (a, b) \to a \circ b = c, c \in G$$

A group has the following properties:

i. **Closed**: $\forall a, b \in G : a \circ b \in G$

ii. **Associative**: $\forall a, b, c \in G : (a \circ b) \circ c = a \circ (b \circ c)$

iii. **Neutral element**: $\exists 0 \in G, \forall a \in G : 0 \circ a = a$

iv. **Inverse elements**: $\forall a \in G, \exists b \in G : a \circ b = 0$

Group Examples

Some easy examples of groups are:

i. $(\mathbb{Z}, +)$

ii. $(\mathbb{Z}_m, +)$ with:
   - $Z_m = \{z \in \mathbb{Z} : z < m\}$
   - $+ : G \times G \to G$ is $+(a, b) = a + b \mod m$

Abelian Group

A group [PP10, AJM96] G is abelian if the operation $\circ$ is abelian (commutative).

So, the properties would be written as:

i. **Closed commutative**: $\forall a, b \in G : a \circ b = b \circ a = c, c \in G$
ii. **Associative**: $\forall a, b, c \in G : (a \circ b) \circ c = a \circ (b \circ c)$
iii. **Neutral element**: $\exists 0 \in G, \forall a \in G : 0 \circ a = a \circ 0 = a$
iv. **Inverse elements**: $\forall a \in G, \exists b \in G : a \circ b = b \circ a = 0$

Ring

A ring [JD20] $(R, +, \times)$ consists of a set R with two operations defined on its elements, here denoted by $+$ and $\times$. For R to qualify as a ring, the operations have to fulfill the following conditions:

i. The structure $(R, +)$ is an Abelian group.
ii. The operation $\times$ is closed, and associative over R. There is a neutral element for $\times$ in R usually denoted by 1.
iii. The two operations $+$ and $\times$ are related by the distributive law: $\forall a, b, c \in R : (a + b) \times c = (a \times c) + (b \times c)$.

A ring $(R, +, \times)$ is called a commutative ring if the operation $\times$ is commutative.

Field

A field [JD20, PP10] F is a set of elements with the following properties:

i. The structure $(F, +, \times)$ is a commutative ring:
   - All elements of F form an additive group with the group operation $+$ and the neutral element 0.
   - All elements of F except for the 0 form a multiplicative group with the operation $\times$ and the neutral element 1.

ii. For all elements of F, there is an inverse element in F with respect to the operation $\times$, except for the element 0, the neutral element of $(F, +)$.

Finite fields

Definition

A field F is a **finite field** (or Galois Field) if the number of elements ($|F|$) is finite.

- The number of elements in the set is called the **order** of the field.
- A field with **order** m exists iff m is a prime power. This is,

$$m = p^n$$

  with $n \in \mathbb{Z}$ and p a prime number.
- p is called the **characteristic** of the finite field.

Finite fields used in the description of Rijndael have characteristic 2.

Finite fields

Remarks

- The previous definition implies there are finite fields with 11 elements or with 169 elements because $169 = 13^2$. But not all *orders* are possible. For example, there's no finite field with 24 elements because $24 = 3 * 2^3$.
- The most intuitive examples of finite fields use $p = 1$ and are called prime fields. In fact, $GF(2)$ plays an important role in Rijndael.
- The notation from now on for a Galois Field will be $GF(p^n)$.

Prime fields

Operations

When the field $GF(p)$ is a prime field we encounter that:

- All nonzero elements of $GF(p)$ have an inverse.
- Arithmetic in $GF(p)$ is done modulo p.

Prime fields

Operations example

Given $GF(5)$ and $a, b \in GF(5)$

$$+(a,b) = a + b \mod 5$$

+	**0**	**1**	**2**	**3**	**4**
0	0	1	2	3	4
1	1	2	3	4	0
2	2	3	4	0	1
3	3	4	0	1	2
4	4	0	1	2	3

$$\times(a,b) = a * b \mod 5$$

$\times$	**0**	**1**	**2**	**3**	**4**
0	0	0	0	0	0
1	0	1	2	3	4
2	0	2	4	1	3
3	0	3	1	4	2
4	0	4	3	2	1

Prime fields

Additive Inverse

Given $GF(5)$ and $a, b \in GF(5)$

$$+(a,b) = a + b \mod 5$$

+	**0**	**1**	**2**	**3**	**4**
0	0	1	2	3	4
1	1	2	3	4	0
2	2	3	4	0	1
3	3	4	0	1	2
4	4	0	1	2	3

$-a$

-0	=	1
-1	=	4
-2	=	3
-3	=	2
-4	=	1

Prime fields

Multiplicative inverse

Given $GF(5)$ and $a, b \in GF(5)$

$$\times(a,b) = a * b \mod 5$$

$\times$	**0**	**1**	**2**	**3**	**4**
0	0	0	0	0	0
1	0	1	2	3	4
2	0	2	4	1	3
3	0	3	1	4	2
4	0	4	3	2	1

a^{-1}

0^{-1}	=	Doesn't exist
1^{-1}	=	1
2^{-1}	=	3
3^{-1}	=	2
4^{-1}	=	4

Prime fields

GF(2)

Given $GF(2)$ and $a, b \in GF(2)$

$$\oplus(a,b) = a + b \mod 2$$

$\oplus$	**0**	**1**
0	0	1
1	1	0

$$\wedge(a,b) = a * b \mod 2$$

$\wedge$	**0**	**1**
0	0	0
1	0	1

Extension Fields

$GF(2^m)$

When m is different than 1. The operations regarding our fields change. In the case of the AES, and for convenient reasons, m is equal to 8.
A field F defined as $GF(2^m)$ will be called an extension field.
Before writing the definition of the multiplication operation over $G(2^8)$ let's introduce the notion of polynomials with coefficients in $G(2^8)$.

Polynomials with coefficients in $G(2^8)$

Definition

In the **AES** algorithm will be presented as the concatenation of its individual bit values (0 or 1) between braces in the order $\{b_7, b_6, b_5, b_4, b_3, b_2, b_1, b_0\}$. As we have 8 bits, we can store it exactly in 1 byte of memory. Let's write these elements using a polynomial representation:

$$b_7x^7 + b_6x^6 + b_5x^5 + b_4x^4 + b_3x^3 + b_2x^2 + b_1x^1 + b_0 = \sum_{i=0}^{7} b_i x^i.$$

For example, $\{01100011\}$ identifies the specific finite field element $x^6 + x^5 + x + 1$.

Polynomials with coefficients in $G(2^8)$

Representation

It is also convenient to denote byte values using hexadecimal notation with each of two groups of four bits being denoted by a single character as followed:

Bit Pattern	Character
0000	0
0001	1
0010	2
0011	3
0100	4
0101	5
0110	6
0111	7

Bit Pattern	Character
1000	8
1001	9
1010	A
1011	B
1100	C
1101	D
1110	E
1111	F

Figure 1. Hexadecimal representation of bit patterns.

Polynomials with coefficients in $G(2^8)$

Example

For example, the following expressions are equivalent to one another:

(polynomial notation);
$(x^6 + x^4 + x^2 + x + 1) + (x^7 + x + 1) = x^7 + x^6 + x^4 + x^2$

(binary notation);
$\{01010111\} \wedge \{10000011\} = \{11010100\}$

(hexadecimal notation);
$\{57\} \wedge \{83\} = \{D4\}$

Polynomials with coefficients in $G(2^8)$

Multiplication

In the polynomial representation, multiplication in $G(2^8)$ (denoted as *bullet*) corresponds to:

Let $a(x), b(x) \in GF(2^8)$

i. Compute $a(x) * b(x)$ as in normal polynomial calculus.
ii. Apply the modulo using an **irreducible polynomial** $m(x)$ of degree 8.

- A polynomial is irreducible if its only divisors are 1 and itself.
- Doing this, we ensure that $a(x) \bullet b(x) \in GF(2^8)$.
- In the AES, $m(x) = x^8 + x^4 + x^3 + x + 1$

Polynomials with coefficients in $G(2^8)$

Multiplication example

For example, $\{57\} \bullet \{83\} = \{C1\}$, because

$$\begin{aligned}(x^6+x^4+x^2+x+1)(x^7+x+1) &= x^{13}+x^{11}+x^9+x^8+x^7+x^7+ \\ &\quad x^5+x^3+x^2+x+x^6+x^4+x^2 \\ &\quad +x+1 \\ &= x^{13}+x^{11}+x^9+x^8+x^6+x^5+ \\ &\quad x^4+x^3+1\end{aligned}$$

and

$$x^{13}+x^{11}+x^9+x^8+x^6+x^5+x^4+x^3+1 \text{ modulo}(x^8+x^4+x^3+x+1) = x^7+x^6+1$$

Polynomials with coefficients in $G(2^8)$

The xtime operation

Multiplying the binary polynomial previously defined with the polynomial x results in

$$b_7x^8 + b_6x^7 + b_5x^6 + b_4x^5 + b_3x^4 + b_2x^3 + b_1x^2 + b_0x.$$

- $x \bullet b(x)$ is obtained by reducing the above result modulo $m(x)$.
- If $b_7 = 0$, the result is already in reduced form.
- If $b_7 = 1$, the reduction is accomplished by subtracting the polynomial $m(x)$.
- It follows that multiplication by x ($\{02\}$) can be implemented at the byte level as a left shift and a subsequent conditional bit wise XOR with $\{1B\}$.

Polynomials with coefficients in $G(2^8)$

Other Example

For example, $\{57\} \bullet \{13\} = \{FE\}$, because

$$\begin{array}{ccccccc}
\{57\} & \bullet & \{02\} & = & xtime(\{57\}) & = & \{AE\} \\
\{57\} & \bullet & \{04\} & = & xtime(\{AE\}) & = & \{47\} \\
\{57\} & \bullet & \{08\} & = & xtime(\{47\}) & = & \{8E\} \\
\{57\} & \bullet & \{10\} & = & xtime(\{8E\}) & = & \{07\}
\end{array}$$

thus,

$$\begin{aligned}
\{57\} \bullet \{13\} &= \{57\} \bullet (\{01\} \oplus \{02\} \oplus \{10\}) \\
&= \{57\} \oplus \{AE\} \oplus \{07\} \\
&= FE
\end{aligned}$$

Polynomials with coefficients in $G(2^8)$

Inversion in $GF(2^8)$

For any non-zero binary polynomial $b(x)$ of degree less than 8, the multiplicative inverse of $b(x)$, denoted $b^{-1}(x)$, can be found as follows: the extended Euclidean algorithm is used to compute polynomials $a(x)$ and $c(x)$ such that

$$b(x)a(x) + m(x)c(x) = 1$$

Hence, $a(x) \bullet b(x) \mod m(x) = 1$, which means

$$b^{-1}(x) = a(x) \mod m(x).$$

Moreover, for any $a(X)$,$b(x)$ and $c(x)$ in the field, it holds that

$$a(x) \bullet (b(x) + c(x)) = a(x) \bullet b(x) + a(x) \bullet c(x)$$

Outline

SubBytes

Definition

- Non-linear byte substitution that operates independently on each byte of the State.
- Constructed by composing two transformations
  1. Take the multiplicative inverse in the finite field GF(2^8); the element $\{00\}$ is mapped to itself.
  2. Apply the following affine transformation (over GF(2)):

     $$b_i' = b_i \oplus b_{(i+4)mod8} \oplus b_{(i+5)mod8} \oplus b_{(i+6)mod8} \oplus b_{(i+7)mod8} \oplus c_i$$

     for $0 \leq i < 8$, where b_i is the i^{th} bit of the byte, and c_i is the i^{th} bit of a byte. c with the value $\{63\}$ or $\{01100011\}$
- Is invertible.

SubBytes

The Affine transformation

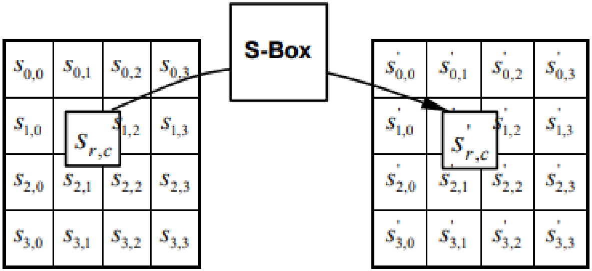
In matrix form, the affine transformation element of the S-box can be expressed as:

$$\begin{bmatrix} b_0' \\ b_1' \\ b_2' \\ b_3' \\ b_4' \\ b_5' \\ b_6' \\ b_7' \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \\ b_6 \\ b_7 \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \end{bmatrix}$$

SubBytes

S-Box Figure

The following figure illustrates the effect of the **SubBytes()** transformation on the State:


S-Box
$s_{0,0}$ $s_{0,1}$ $s_{0,2}$ $s_{0,3}$
$s_{1,0}$ $s_{r,c}$ $s_{1,2}$ $s_{1,3}$
$s_{2,0}$ $s_{2,1}$ $s_{2,2}$ $s_{2,3}$
$s_{3,0}$ $s_{3,1}$ $s_{3,2}$ $s_{3,3}$
$s'_{0,0}$ $s'_{0,1}$ $s'_{0,2}$ $s'_{0,3}$
$s'_{1,0}$ $s'_{r,c}$ $s'_{1,2}$ $s'_{1,3}$
$s'_{2,0}$ $s'_{2,1}$ $s'_{2,2}$ $s'_{2,3}$
$s'_{3,0}$ $s'_{3,1}$ $s'_{3,2}$ $s'_{3,3}$

Sbox

The S-box used in the **SubBytes()** transformation is presented in hexadecimal for as in figure.
For example, if $s_{1,1} = 53$, then the substitution value would be determined by the intersection of the row with index '5' and the column with index '3'. This would result in $s^{'}_{1,1}$ having a value of *ED*.

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	0x63	0x7C	0x77	0x7B	0xF2	0x6B	0x6F	0xC5	0x30	0x01	0x67	0x2B	0xFE	0xD7	0xAB	0x76
1	0xCA	0x82	0xC9	0x7D	0xFA	0x59	0x47	0xF0	0xAD	0xD4	0xA2	0xAF	0x9C	0xA4	0x72	0xC0
2	0xB7	0xFD	0x93	0x26	0x36	0x3F	0xF7	0xCC	0x34	0xA5	0xE5	0xF1	0x71	0xD8	0x31	0x15
3	0x04	0xC7	0x23	0xC3	0x18	0x96	0x05	0x9A	0x07	0x12	0x80	0xE2	0xEB	0x27	0xB2	0x75
4	0x09	0x83	0x2C	0x1A	0x1B	0x6E	0x5A	0xA0	0x52	0x3B	0xD6	0xB3	0x29	0xE3	0x2F	0x84
5	0x53	0xD1	0x00	0xED	0x20	0xFC	0xB1	0x5B	0x6A	0xCB	0xBE	0x39	0x4A	0x4C	0x58	0xCF
6	0xD0	0xEF	0xAA	0xFB	0x43	0x4D	0x33	0x85	0x45	0xF9	0x02	0x7F	0x50	0x3C	0x9F	0xA8
7	0x51	0xA3	0x40	0x8F	0x92	0x9D	0x38	0xF5	0xBC	0xB6	0xDA	0x21	0x10	0xFF	0xF3	0xD2
8	0xCD	0x0C	0x13	0xEC	0x5F	0x97	0x44	0x17	0xC4	0xA7	0x7E	0x3D	0x64	0x5D	0x19	0x73
9	0x60	0x81	0x4F	0xDC	0x22	0x2A	0x90	0x88	0x46	0xEE	0xB8	0x14	0xDE	0x5E	0x0B	0xDB
A	0xE0	0x32	0x3A	0x0A	0x49	0x06	0x24	0x5C	0xC2	0xD3	0xAC	0x62	0x91	0x95	0xE4	0x79
B	0xE7	0xC8	0x37	0x6D	0x8D	0xD5	0x4E	0xA9	0x6C	0x56	0xF4	0xEA	0x65	0x7A	0xAE	0x08
C	0xBA	0x78	0x25	0x2E	0x1C	0xA6	0xB4	0xC6	0xE8	0xDD	0x74	0x1F	0x4B	0xBD	0x8B	0x8A
D	0x70	0x3E	0xB5	0x66	0x48	0x03	0xF6	0x0E	0x61	0x35	0x57	0xB9	0x86	0xC1	0x1D	0x9E
E	0xE1	0xF8	0x98	0x11	0x69	0xD9	0x8E	0x94	0x9B	0x1E	0x87	0xE9	0xCE	0x55	0x28	0xDF
F	0x8C	0xA1	0x89	0x0D	0xBF	0xE6	0x42	0x68	0x41	0x99	0x2D	0x0F	0xB0	0x54	0xBB	0x16

InvSubBytes

Definition

- The S-Box can be inversed by applying the inverse of each operation in reverse order.
- In the S-Box computation we first applied the Galois Field inverse and then the affine function.
- We will apply the inverse of the affine function (which is also affine) and then the inverse of the Galois field which is the function itself.
- The derivations of the inverse of the affine function are vaguely specified in the bibliography consulted.
- As in the SubBytes operation, we will try to build a lookup table.

InvSubBytes

The inverse Affine transformation

In matrix form, the affine transformation element of the S-box can be expressed as:

$$\begin{bmatrix} b_0' \\ b_1' \\ b_2' \\ b_3' \\ b_4' \\ b_5' \\ b_6' \\ b_7' \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \\ b_6 \\ b_7 \end{bmatrix} + \begin{bmatrix} 1 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

InvSbox

The Inv-S-box used in the **InvSubBytes()** transformation is presented in hexadecimal for as in figure.
For example, if $s_{1,1} = 53$, then the substitution value would be determined by the intersection of the row with index '5' and the column with index '3'. This would result in $s^{'}_{1,1}$ having a value of 50.

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	0x52	0x09	0x6A	0xD5	0x30	0x36	0xA5	0x38	0xBF	0x40	0xA3	0x9E	0x81	0xF3	0xD7	0xFB
1	0x7C	0xE3	0x39	0x82	0x9B	0x2F	0xFF	0x87	0x34	0x8E	0x43	0x44	0xC4	0xDE	0xE9	0xCB
2	0x54	0x7B	0x94	0x32	0xA6	0xC2	0x23	0x3D	0xEE	0x4C	0x95	0x0B	0x42	0xFA	0xC3	0x4E
3	0x08	0x2E	0xA1	0x66	0x28	0xD9	0x24	0xB2	0x76	0x5B	0xA2	0x49	0x6D	0x8B	0xD1	0x25
4	0x72	0xF8	0xF6	0x64	0x86	0x68	0x98	0x16	0xD4	0xA4	0x5C	0xCC	0x5D	0x65	0xB6	0x92
5	0x6C	0x70	0x48	0x50	0xFD	0xED	0xB9	0xDA	0x5E	0x15	0x46	0x57	0xA7	0x8D	0x9D	0x84
6	0x90	0xD8	0xAB	0x00	0x8C	0xBC	0xD3	0x0A	0xF7	0xE4	0x58	0x05	0xB8	0xB3	0x45	0x06
7	0xD0	0x2C	0x1E	0x8F	0xCA	0x3F	0x0F	0x02	0xC1	0xAF	0xBD	0x03	0x01	0x13	0x8A	0x6B
8	0x3A	0x91	0x11	0x41	0x4F	0x67	0xDC	0xEA	0x97	0xF2	0xCF	0xCE	0xF0	0xB4	0xE6	0x73
9	0x96	0xAC	0x74	0x22	0xE7	0xAD	0x35	0x85	0xE2	0xF9	0x37	0xE8	0x1C	0x75	0xDF	0x6E
A	0x47	0xF1	0x1A	0x71	0x1D	0x29	0xC5	0x89	0x6F	0xB7	0x62	0x0E	0xAA	0x18	0xBE	0x1B
B	0xFC	0x56	0x3E	0x4B	0xC6	0xD2	0x79	0x20	0x9A	0xDB	0xC0	0xFE	0x78	0xCD	0x5A	0xF4
C	0x1F	0xDD	0xA8	0x33	0x88	0x07	0xC7	0x31	0xB1	0x12	0x10	0x59	0x27	0x80	0xEC	0x5F
D	0x60	0x51	0x7F	0xA9	0x19	0xB5	0x4A	0x0D	0x2D	0xE5	0x7A	0x9F	0x93	0xC9	0x9C	0xEF
E	0xA0	0xE0	0x3B	0x4D	0xAE	0x2A	0xF5	0xB0	0xC8	0xEB	0xBB	0x3C	0x83	0x53	0x99	0x61
F	0x17	0x2B	0x04	0x7E	0xBA	0x77	0xD6	0x26	0xE1	0x69	0x14	0x63	0x55	0x21	0x0C	0x7D

Outline

```c
typedef unsigned char byte;

byte mult(byte in_1, byte in_2){
  byte mask=0x01, result=0x00, piv=in_1;

  for(int i=0; i<8; i++){
    if(in_2 & mask) result^=piv;
    mask = mask << 1;
    piv = xtime(piv);
  }

  return result;
}
```

Listing 1: Mult example

References (1)

- Scott A Vanstone Alfred J. Menezes, *Handbook of applied cryptography*, CRC Press, 1996.
- Vincent Rijmen Joan Daemen, *The design of rijndael: The advanced encryption standard (aes)*, 2nd edition ed., Information Security And Cryptography, Springer, 2020.
- J. Katz and Y. Lindell, *Introduction to modern cryptography*, Chapman & Hall/CRC Cryptography and Network Security Series, Taylor & Francis, 2014.
- Christof Paar and Jan Pelzl, *Understanding cryptography: A textbook for students and practitioners*, Springer, 2010.

References (2)

D. Venturi, *Crittografia nel paese delle meraviglie*, UNITEXT, Springer Milan, 2012.