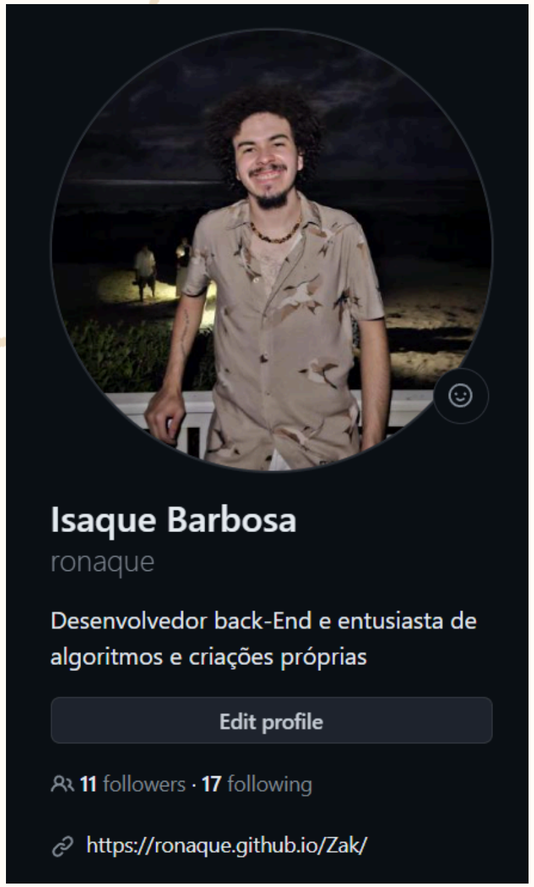
O Poder do XOR na criptografia

Por Isaque Barbosa

Isaque



Técnico em Jogos Digitais pelo IFRN
Graduando em Tecnologia da Informação pela UFRN
Aspirante a cientista da computação com foco em algoritmos criptográficos
Trabalhei 1 ano com Python no desenvolvimento Back-End
E esta é minha primeira palestra! 🫠

Tópicos

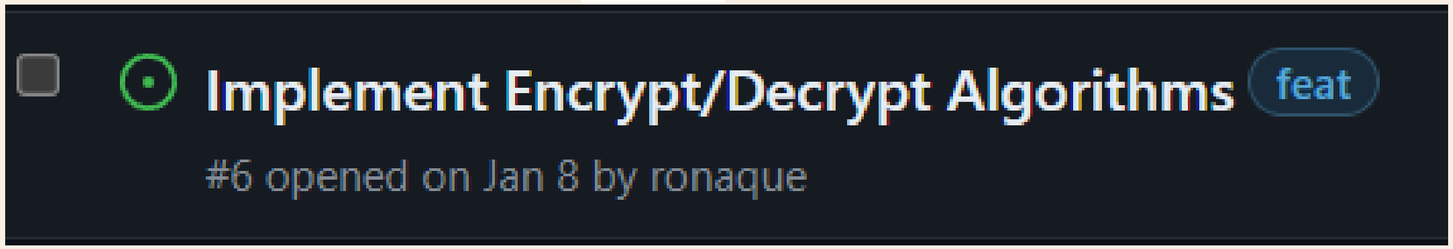
1. Motivação
2. O XOR
3. O XOR na criptografia
4. RC5
5. AES
6. Saiba mais
7. Links

+

Faça sua própria criptografia, só por curiosidade

— Akita On Rails

=



O XOR

Oi, eu sou se amiguinho
XOR
ou simplesmente

^

Eu sou um operador booliano
Ou seja, opero com 0 ou 1

De bit em bit
Veja bem

x	y	x ^ y
0	0	0
0	1	1
1	0	1
1	1	0

Me veja em execução!

```
x = 0 y = 0 x ^ y = 0
x = 0 y = 1 x ^ y = 1
x = 1 y = 0 x ^ y = 1
x = 1 y = 1 x ^ y = 0
```

E números com vários bits?
Eu vou de bit em bit!

```
x = 9 y = 5 x ^ y = 12
        Veja bem:
        x = 1001
        y = 0101
    x ^ y = 1100
```

O XOR na Criptografia

A criptografia envolve a **conversão de texto simples e legível em texto incompreensível**, o que é conhecido como texto cifrado.

O XOR na Criptografia

A criptografia envolve a **conversão de texto simples e legível em texto incompreensível**, o que é conhecido como texto cifrado.

Ta, mas onde o XOR se relaciona com texto?

O XOR na Criptografia

A criptografia envolve a **conversão de texto simples e legível em texto incompreensível**, o que é conhecido como texto cifrado.

Ta, mas onde o XOR se relaciona com texto?

ASCII TABLE

Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char
0	0	[NULL]	32	20	[SPACE]	64	40	@	96	60	`
1	1	[START OF HEADING]	33	21	!	65	41	A	97	61	a
2	2	[START OF TEXT]	34	22	"	66	42	B	98	62	b
3	3	[END OF TEXT]	35	23	#	67	43	C	99	63	c
4	4	[END OF TRANSMISSION]	36	24	$	68	44	D	100	64	d
5	5	[ENQUIRY]	37	25	%	69	45	E	101	65	e
6	6	[ACKNOWLEDGE]	38	26	&	70	46	F	102	66	f
7	7	[BELL]	39	27	'	71	47	G	103	67	g
8	8	[BACKSPACE]	40	28	(	72	48	H	104	68	h
9	9	[HORIZONTAL TAB]	41	29	)	73	49	I	105	69	i
10	A	[LINE FEED]	42	2A	*	74	4A	J	106	6A	j
11	B	[VERTICAL TAB]	43	2B	+	75	4B	K	107	6B	k
12	C	[FORM FEED]	44	2C	,	76	4C	L	108	6C	l
13	D	[CARRIAGE RETURN]	45	2D	-	77	4D	M	109	6D	m
14	E	[SHIFT OUT]	46	2E	.	78	4E	N	110	6E	n
15	F	[SHIFT IN]	47	2F	/	79	4F	O	111	6F	o
16	10	[DATA LINK ESCAPE]	48	30	0	80	50	P	112	70	p
17	11	[DEVICE CONTROL 1]	49	31	1	81	51	Q	113	71	q
18	12	[DEVICE CONTROL 2]	50	32	2	82	52	R	114	72	r
19	13	[DEVICE CONTROL 3]	51	33	3	83	53	S	115	73	s
20	14	[DEVICE CONTROL 4]	52	34	4	84	54	T	116	74	t
21	15	[NEGATIVE ACKNOWLEDGE]	53	35	5	85	55	U	117	75	u
22	16	[SYNCHRONOUS IDLE]	54	36	6	86	56	V	118	76	v
23	17	[END OF TRANS. BLOCK]	55	37	7	87	57	W	119	77	w
24	18	[CANCEL]	56	38	8	88	58	X	120	78	x
25	19	[END OF MEDIUM]	57	39	9	89	59	Y	121	79	y
26	1A	[SUBSTITUTE]	58	3A	:	90	5A	Z	122	7A	z
27	1B	[ESCAPE]	59	3B	;	91	5B	[	123	7B	{
28	1C	[FILE SEPARATOR]	60	3C	<	92	5C	\	124	7C	|
29	1D	[GROUP SEPARATOR]	61	3D	=	93	5D	]	125	7D	}
30	1E	[RECORD SEPARATOR]	62	3E	>	94	5E	^	126	7E	~
31	1F	[UNIT SEPARATOR]	63	3F	?	95	5F	_	127	7F	[DEL]

O XOR na Criptografia

A criptografia envolve a **conversão de texto simples e legível em texto incompreensível**, o que é conhecido como texto cifrado.

Tá, mas onde o XOR se relaciona com texto?

```
Vamos me ver com chars!

x = 0 y = i

ord(x) = 79
bin(ord(x)) = 0b1001111

ord(y) = 105
bin(ord(y)) = 0b1101001

Então podemos ter x ^ y!

x = 0b1001111
y = 0b1101001
x ^ y = 0b0100110
x ^ y = &
```

O XOR na Criptografia

A criptografia envolve a **conversão de texto simples e legível em texto incompreensível**, o que é conhecido como texto cifrado.

Tá, mas onde o XOR se relaciona com texto?

Massa! Já sei como tornar meu texto incompreensível, mas como meu destinatário vai ler?

```
Note que se um dos meus
  Operandos for fixo
  Você pode voltar atrás!
Eu sou inversa de mim mesma!
```

```
x =  0, y = i
w = x ^ y = &
z = w ^ y = 0
```

```
  Então com um texto fixo
       (Uma chave)
O remetente e o destinatário
Leem o mesmo texto original!
```

```
x =  Hello Grupy
y =  XOR python!
w = x ^ y = *>LY3X
z = w ^ y = Hello Grupy
```

Fortalecendo o uso do XOR

RC5

```
Enter the 128 bits key (16 characters) to be used in the encryption: eai galera grupy
Enter the text to be encrypted: eu sou o isaque!
Initializing encryption of eu sou o isaque! with the key eai galera grupy
Encrypting 0-8 characters of eu sou o isaque!: [1702174835, 1869946991]
Created the Expanded Key Table S with size equals to 2 * (number of rounds(12) + 1)
Expanded Key Table S:
[2586377890, 3073491602, 2614257280, 3550296819,
4273085636, 2660731107, 1197185952, 3880555570,
3640780025, 414818278, 1394802885, 3224181948,
3402184260, 3807338997, 1614146604, 1614480263,
3300884603, 1279187747, 2678687763, 3567713409,
589498198, 4260189572, 67070184, 3247681227,
892548254, 1047438560]

Encrypting 8-16 characters of eu sou o isaque!: [543781729, 1903519009]

Plaintext:
eu sou o isaque!
Ciphertext:
Gö2ä«:è(i{»Ú9
Plaintext Decrypted:
eu sou o isaque!
```

Cria-se uma tabela de valores baseados na chave, onde esta tabela tem tamanho = 2 * (O número de rodadas + 1)

Os quatro primeiros caracteres de cada partição da palavra será encriptado com os valores impares, e os quatro últimos com os pares

RC5

```
A = to_four_bytes(input[0] + S[0])
B = to_four_bytes(input[1] + S[1])

i = 1
while i <= r:
    A = to_four_bytes((left_rotate(A ^ B, B)) + S[2*i])
    B = to_four_bytes((left_rotate(B ^ A, A)) + S[2*i+1])
    i += 1

output[0] = A
output[1] = B
```

É utilizado uma função chamada left rotate, onde um de seus parâmetros recebe o resultado de uma operação XOR!

RC5

```
A = to_four_bytes(input[0])
B = to_four_bytes(input[1])

i = r
while i > 0:
    B = (right_rotate(ct.c_uint32(B - S[2*i+1]).value, A)) ^ A
    A = right_rotate(ct.c_uint32(A - S[2*i]).value, B) ^ B
    i -= 1

output[0] = ct.c_uint32(A - S[0]).value
output[1] = ct.c_uint32(B - S[1]).value
```

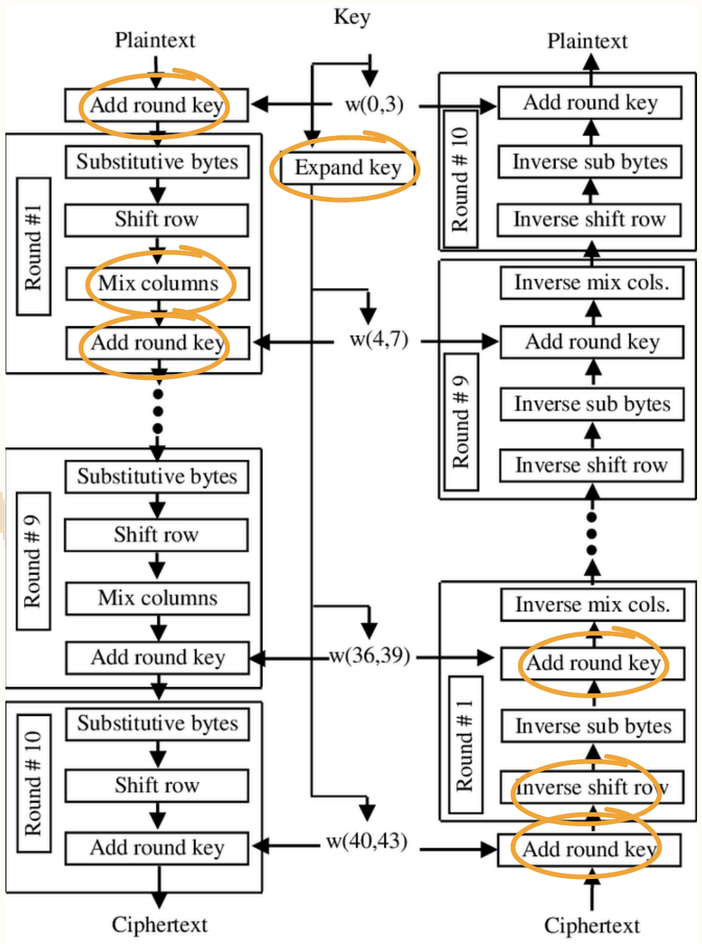
Decrypt

```
A = to_four_bytes(input[0] + S[0])
B = to_four_bytes(input[1] + S[1])

i = 1
while i <= r:
    A = to_four_bytes((left_rotate(A ^ B, B)) + S[2*i])
    B = to_four_bytes((left_rotate(B ^ A, A)) + S[2*i+1])
    i += 1

output[0] = A
output[1] = B
```

Encrypt

Por fim, para a decriptação, note que é realizado o caminho contrário da encriptação justamente por causa do poder do XOR!

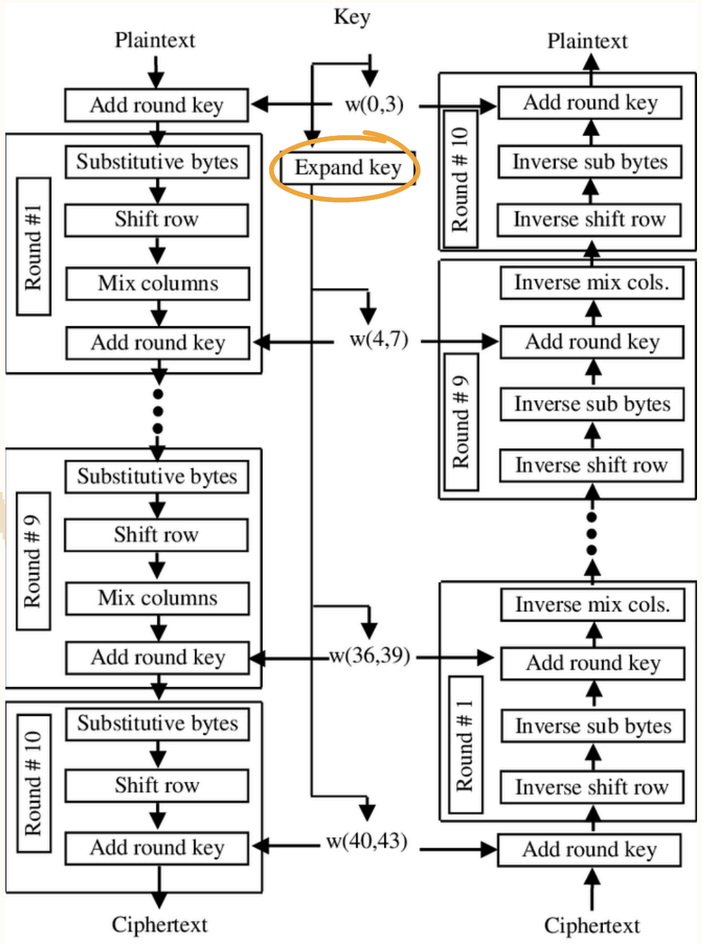
AES



O AES é o principal algoritmo de encriptação/decriptação atualmente.

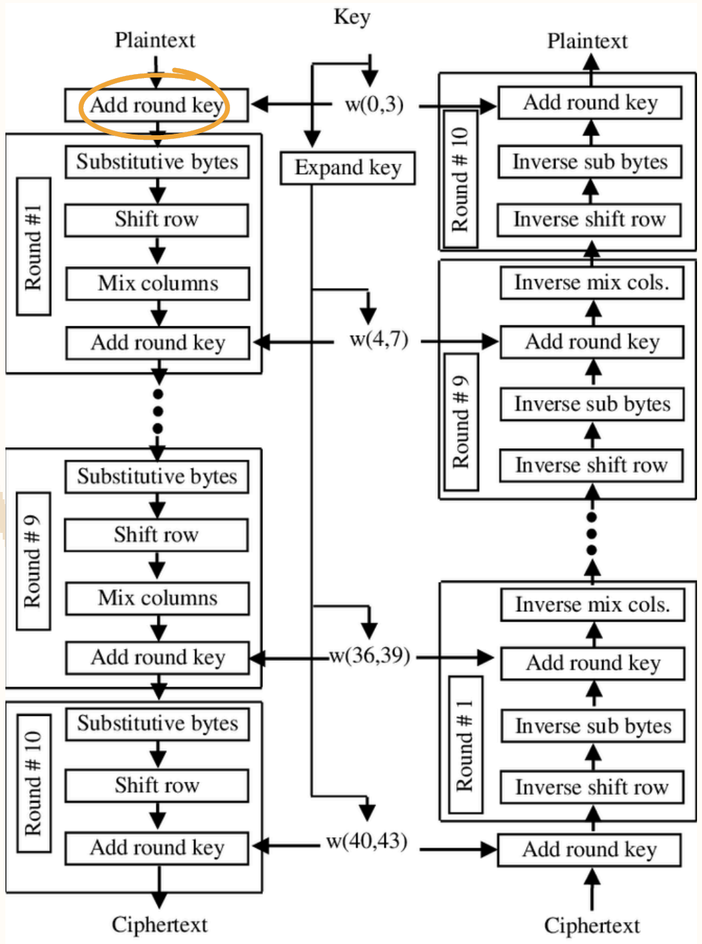
Nos pontos marcados, pode se ver onde é utilizado o operador XOR!

AES




```python
def _expand_key(self, master_key):
    """
    Expands and returns a list of key matrices for the given master_key.
    """
    # Initialize round keys with raw key material.
    key_columns = bytes2matrix(master_key)
    iteration_size = len(master_key) // 4

    i = 1
    while len(key_columns) < (self.n_rounds + 1) * 4:
        # Copy previous word.
        word = list(key_columns[-1])

        # Perform schedule_core once every "row".
        if len(key_columns) % iteration_size == 0:
            # Circular shift.
            word.append(word.pop(0))
            # Map to S-BOX.
            word = [s_box[b] for b in word]
            # XOR with first byte of R-CON, since the others bytes of R-CON
            are 0.
            word[0] ^= r_con[i]
            i += 1
        elif len(master_key) == 32 and len(key_columns) % iteration_size == 4:
            # Run word through S-box in the fourth iteration when using a
            # 256-bit key.
            word = [s_box[b] for b in word]

        # XOR with equivalent word from previous iteration.
        word = xor_bytes(word, key_columns[-iteration_size])
        key_columns.append(word)

    # Group key words in 4x4 byte matrices.
    return [key_columns[4*i : 4*(i+1)] for i in range(len(key_columns) // 4)]
```

AES

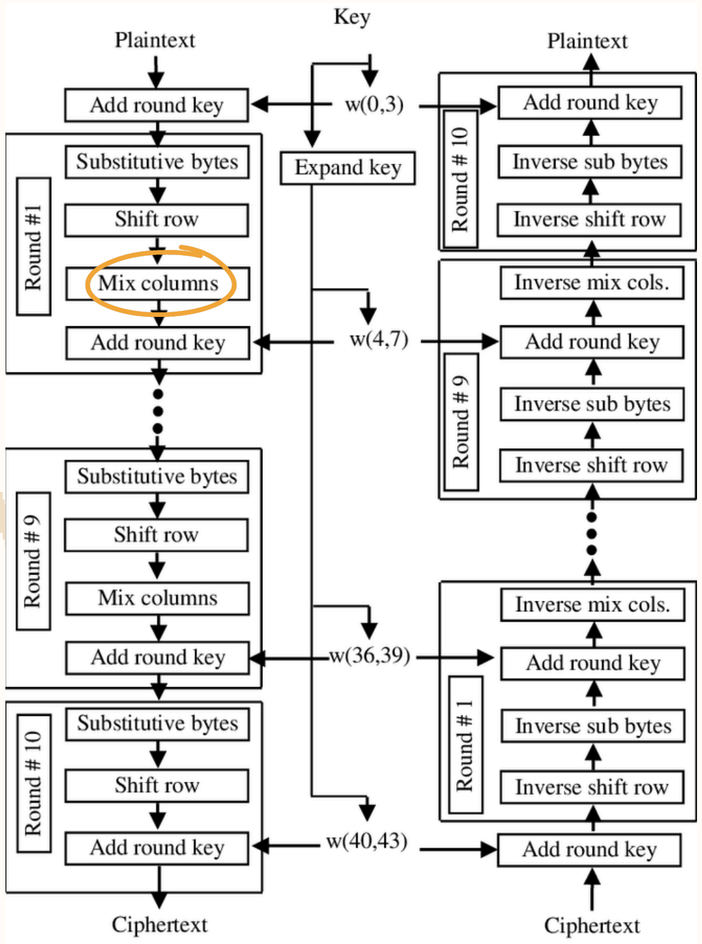



```python
def add_round_key(s, k):
    for i in range(4):
        for j in range(4):
            s[i][j] ^= k[i][j]
```

S é o estado atual do texto formatado em uma Matriz

K é uma matriz com a expansão da Chave original

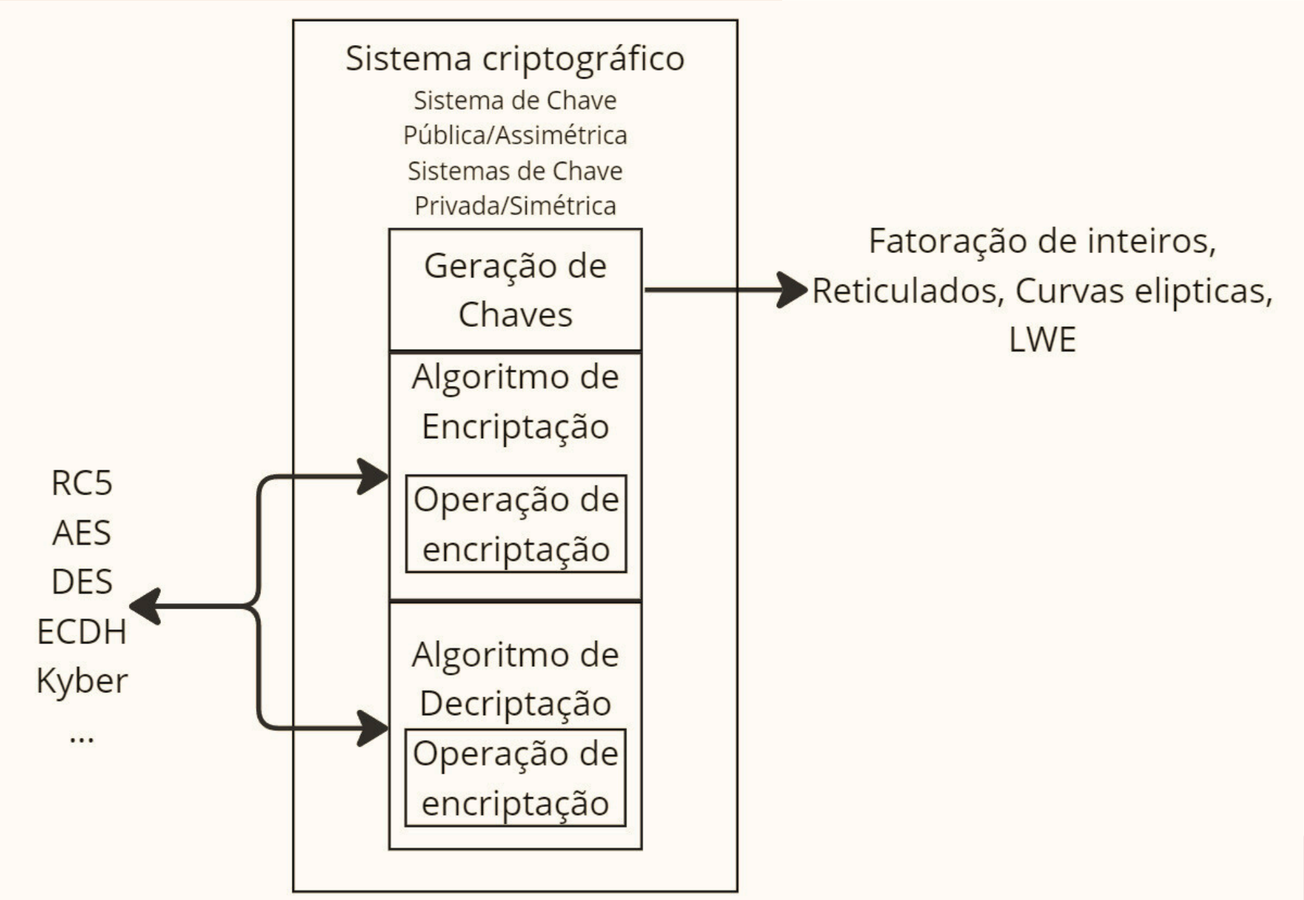
AES




```python
def mix_single_column(a):
    # see Sec 4.1.2 in The Design of Rijndael
    t = a[0] ^ a[1] ^ a[2] ^ a[3]
    u = a[0]
    a[0] ^= t ^ xtime(a[0] ^ a[1])
    a[1] ^= t ^ xtime(a[1] ^ a[2])
    a[2] ^= t ^ xtime(a[2] ^ a[3])
    a[3] ^= t ^ xtime(a[3] ^ u)


def mix_columns(s):
    for i in range(4):
        mix_single_column(s[i])
```

Para a matriz “s”, irá misturar os valores de cada coluna “a”

Saiba Mais!


Sistema criptográfico
Sistema de Chave Pública/Assimétrica
Sistemas de Chave Privada/Simétrica
Geração de Chaves
Algoritmo de Encriptação
Operação de encriptação
Algoritmo de Decriptação
Operação de encriptação
Fatoração de inteiros, Reticulados, Curvas elipticas, LWE
RC5
AES
DES
ECDH
Kyber
...

Links

Meu Github + LinkedIn e Insta

Código RC5

RC5 Paper

Código AES

AES Paper

/ronaque

@isaquebarbosam

Obrigado!