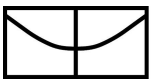


Universidade de Brasília

Instituto de Ciências Exatas
Departamento de Ciência da Computação

Trabalho 2

Segurança Computacional

Guilherme Braga Pinto | 170162290
Gabriel Carvalho Moretto | 150154917

Brasília
2021

Introdução

Este trabalho tem por objetivo dar suporte à cifra AES de 128 bits, em modo simples e em modo contador, para uma quantidade “x” possível de repetições para cada modo. A partir de uma imagem, geramos uma string com seu arquivo para servir de input para a cifragem. Essa string de números hexadecimais é convertida em uma matriz 4x4, assim como a key que deve ser providenciada. O ato de decifrar ocorre com uma string de input e uma chave de 32 caracteres representando os 128 bits em hexadecimal necessários para o programa funcionar. Dados foram tratados como listas de listas (matriz 4x4) contendo strings, sendo convertidos de string (dado em hexadecimal) para inteiro ao se executar operações matemáticas. O resultado das operações podem ser exportados para um arquivo de texto, e o programa mostra sua primeira execução para demonstrar modularmente o que ocorre a cada etapa de cifragem e decifragem. Este trabalho foi realizado em Python, em ambiente Linux Ubuntu 18.04. Para rodar, execute:

python3 main.py

O modo de operação ECB faz a cifragem e decifragem de maneira simples, de 32 em 32 caracteres que representam uma porção daquele número em hexadecimal. A imagem de teste utilizada foi a *lenna.png.*

Funções Principais

As principais funções do programa são:

- *Add Round Key*: Função que realiza operação *XOR* entre duas matrizes.
- *Sub Bytes* (junto de sua função inversa): A partir de uma matriz de valores em hexadecimal *S-BOX*, pegamos um novo elemento na matriz de acordo com o valor dado de input. Os 8 bits mais significativos correspondem à linha dessa matriz, e os 8 bits menos significativos representam a coluna da matriz *S-BOX*. Assim, deve-se realizar a substituição de acordo. A função inversa funciona da mesma maneira, apenas com a *S-BOX* invertida.
- *Shift Rows* (junto de sua função inversa): Essa função realiza o shift à esquerda para cada linha da matriz. A primeira linha da matriz permanece igual, a segunda linha passa por 1 *shift*, a terceira linha passa por 2 *shifts* e a quarta linha passa por 3 *shifts*.
- *Mix Columns* (junto de sua função inversa): A função *Mix Columns* foi simplificada seguindo o seguinte guia. A multiplicação foi dividida por etapas, nas quais cada item de cada coluna deve ser multiplicado pela constante correta de maneira a respeitar a aritmética de campo finito $GF(2^8)$. Dessa forma, a fim de demonstrar como exemplo com uma matriz de $resultado[4]$ obtemos:

$$resultado[0] = multiplica_2(input[0]) \oplus multiplica_3(input[1]) \oplus input[2] \oplus input[3]$$

$$resultado[1] = multiplica_2(input[1]) \oplus multiplica_3(input[2]) \oplus input[3] \oplus input[0]$$

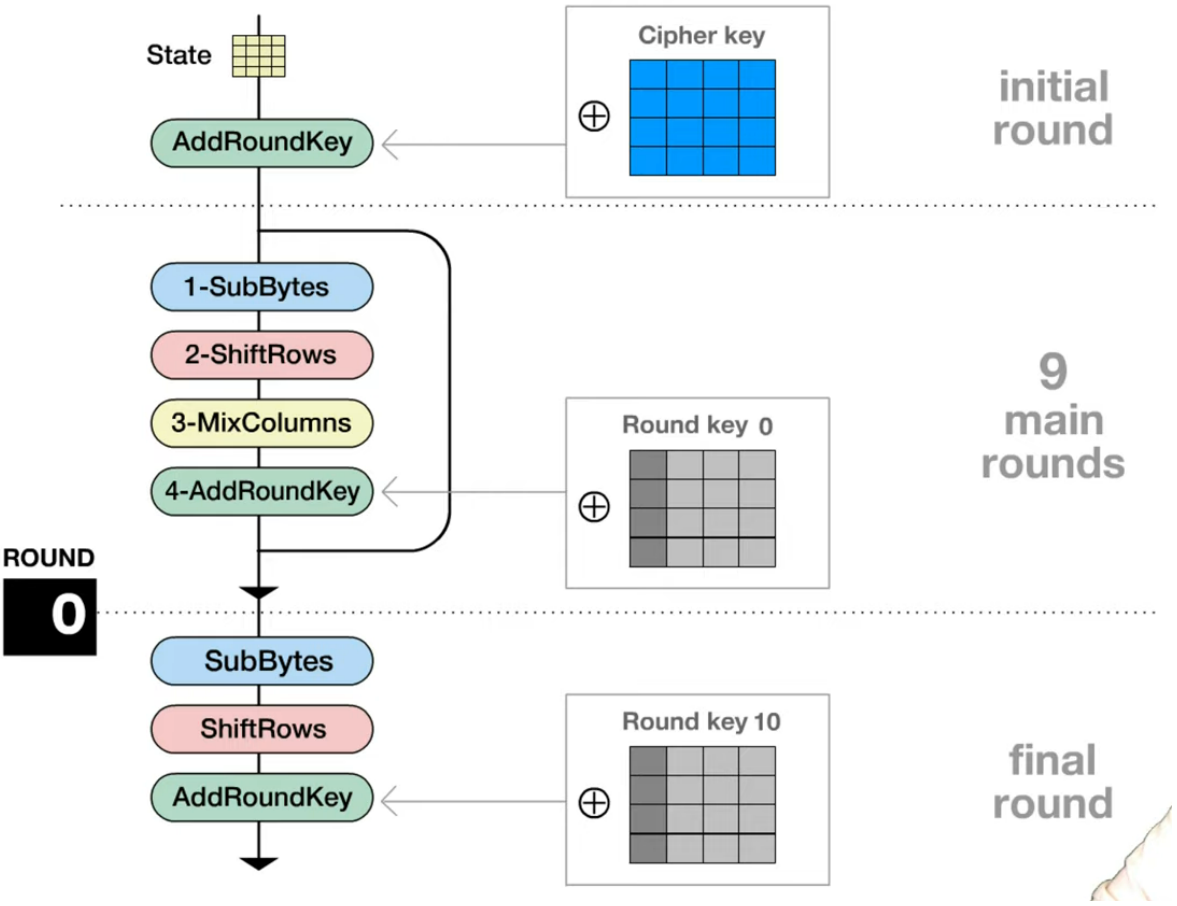
$$resultado[2] = multiplica_2(input[2]) \oplus multiplica_3(input[3]) \oplus input[0] \oplus input[1]$$

$$resultado[3] = multiplica_2(input[3]) \oplus multiplica_3(input[0]) \oplus input[1] \oplus input[2]$$

- A função *Mix Columns* inversa funciona realizando essa multiplicação 3 vezes seguidas.
- A cifragem/decifragem ocorre de dois modos: **ECB** E **CTR**. O modo ECB é a cifragem simples que ocorre a cada 128 bits de um dado input, exatamente "*x*" vezes (sendo "*x*" um número dado pelo usuário). No final da cifragem e decifragem de cada segmento, o resultado é concatenado em uma string de resultado. O modo CTR ocorre de maneira análoga, com a diferença de que no começo de cada interação se realiza um *XOR* com a entrada e um número que representa um contador que incrementa a cada bloco de 128 bits sendo cifrado. A cifragem ocorre exatamente "*x*" vezes para cada segmento (sendo "*x*" um número dado pelo usuário).

Resultados

A cifragem e a decifragem seguem o diagrama a seguir:



fonte: https://www.youtube.com/watch?v=lnKPoWZnNNM

```
Que operação realizar?
Responda com número correspondente.

1- Cifrar (AES ECB)
2- Decifrar(AES ECB)
3- Cifrar (AES CTR)
4- Decifrar (AES CTR)
1
Entre com o nome da imagem.
Exemplo: 'lenna.png'
lenna.png
Entre com a sua chave
Espera-se 32 caracteres (HEX)
5a746f2a4f6e15202e696e642054a46f
Entre com o número de repetições desejadas na cifragem!
1


Passo a passo, exemplo para primeira iteração da primeira parte da string
Key:
[['5a' '4f' '2e' '20']
 ['74' '6e' '69' '54']
 ['6f' '15' '6e' 'a4']
 ['2a' '20' '64' '6f']]
Matriz atual:
[['89' '0d' '00' '49']
 ['50' '0a' '00' '48']
 ['4e' '1a' '00' '44']
 ['47' '0a' '0d' '52']]
Add round key
[['d3' '42' '2e' '69']
 ['24' '64' '69' '1c']
 ['21' 'f' '6e' 'e0']
 ['6d' '2a' '69' '3d']]
Sub bytes
[['66' '2c' '31' 'f9']
 ['36' '43' 'f9' '9c']
 ['fd' '76' '9f' 'e1']
 ['3c' 'e5' 'f9' '27']]
Shift rows
[['66' '2c' '31' 'f9']
 ['43' 'f9' '9c' '36']
 ['9f' 'e1' 'fd' '76']
 ['27' '3c' 'e5' 'f9']]
Mix columns
[['c' 'a' '11' 'b']
 ['16' '44' 'e' '30']
 ['c' '2f' '4e' '1a']
 ['15' '12' 'b' '41']]
Deseja salvar o resultado em um arquivo de texto?[Y/N]?

n
```

Resultado da execução do código para uma cifragem ECB.

Considerações Finais

Muito tempo foi gasto em partes do trabalho que acabam não sendo muito relacionadas com os conceitos do AES de 128 bits. Devido à gestão de tempo e de imprevistos por razões de saúde do Gabriel Moretto, não foi possível completar a demonstração da cifragem e da decifragem da imagem, assim focou-se em demonstrar os conceitos mais relevantes do AES.