

# PROCESSO DE DECRIPTAÇÃO DO ARQUIVO “arquivo-weak-9.in-full.hex”

Aluno: Pedro Lucas Moraes de Sousa Rosa

Professor: Wewerton Luis da Costa Cordeiro

A implementação utilizada foi um ataque de força bruta focado e otimizado, baseado em algumas informações conhecidas:

1ª Informação conhecida critica:

- A chave começa com "SecurityAES".
- O texto decifrado é legível em ASCII.
- O algoritmo usado é o AES no modo ECB.

2ª Estratégia de força bruta otimizada:

- Em vez de testar todas as combinações possíveis de 16 bytes, o código:
  - Usa o prefixo conhecido "SecurityAES".
  - Precisa descobrir apenas os caracteres restantes.
  - Divide o espaço de busca em grupos menores de caracteres:

```
caracteres1 = "abcdefghi"  
caracteres11 = "jklmnopqr"  
caracteres12 = "stuvwxyz"  
caracteres2 = "ABCDEFGHI"  
caracteres21 = "JKLMNOPQR"  
caracteres22 = "STUVWXYZ"  
caracteres3 = "01234"  
caracteres31 = "56789"
```

3ª Paralelização do Processamento:

- Usa a biblioteca "multiprocessing" com POOL.
- Cada processo trabalha com um subconjunto diferente do espaço de busca.
- O código testa diferentes combinações simultaneamente através de "pool.apply\_async".

#### 4º Validação Eficiente:

```
def is_texto_legivel(texto):  
    try:  
        v1 = all(ord(c) < 128 for c in texto.decode('ascii'))  
        return v1  
    except:  
        return False
```

- Verifica rapidamente se o texto decifrado é ASCII legível.
- Funciona como critério de parada quando encontra a chave correta.

#### 5º Processo de Decifração:

```
def tenta_decifrar(texto_cifrado, chave):  
    cifra = AES.new(chave, AES.MODE_ECB)  
    try:  
        decifrado = cifra.decrypt(bytes.fromhex(texto_cifrado))  
        return decifrado  
    except:  
        return None
```

- Usa a biblioteca "Cryptodome" para implementação do AES.
- Modo ECB permite testar cada chave independentemente.

#### 6º Monitoramento e Logging:

- Registra progresso a cada 60 segundos.
- Mantém estatísticas de performance:
  - Número de chaves testadas.
  - Tempo decorrido.
  - Taxa de tentativas por segundo.

Está implementação é particularmente eficiente porque:

1º - Reduz drasticamente o espaço de busca usando o prefixo conhecido.

2º - Paraleliza o processamento para aumentar a velocidade.

3º - Tem um critério claro de sucesso (texto ASCII legível).

4º - Divide o problema em partes menores e gerenciáveis.

Resultado do arquivo decriptado:

- Chave = SecurityAES3LUbU.
- Código Secreto do arquivo = nXzDkn.
- Tempo máximo de decriptação = 510.86 segundos (8 minutos e 514 segundos)
- Quantidade de chaves testadas até encontrar a chave correta: 53.324.077.
- Média de chaves testadas por segundo: 835.049,68 (8 threads) chaves/seg.

```
decifrado_weak_FINAL_2.txt
1 Cras scelerisque pellentesque lectus, quis varius risus varius non.
2 Fusce eu augue at ante bibendum mollis. Quisque accumsan dapibus purus, id sodales dolor. Morbi feugiat tristique facilisis.
3 Maecenas nisl velit, gravida et nisl mattis, euismod interdum lacus. Quisque et est iaculis, malesuada tortor eget, dignissim justo. Nulla ut sodales ex.
4 Suspendisse at vestibulum velit. In hac habitasse platea dictumst. Proin facilisis faucibus tellus, vitae sodales est porta sit amet. Nullam eget accumsan dolor.
5 In feugiat nunc sed nunc pretium, non bibendum ante condimentum. Vestibulum ac ultrices nisl. Praesent arcu tortor, vestibulum et dolor in, pretium sollicitudin ipsum.
6 Maecenas vitae ipsum quis ligula sagittis egestas quis ut mauris. Mauris lacus metus, accumsan sit amet consequat id, rutrum vel neque. Nunc accumsan laoreet justo.
7 In hac habitasse platea dictumst. Nullam viverra, tellus ullamcorper mattis vulputate, lectus mauris aliquam justo, a ultricies felis ipsum ut tellus.
8 Maecenas ac dui non diam malesuada posuere. Cras lorem est, molestie at est nec, varius tempor est. Cras maximus maximus nunc. Aliquam sagittis metus id maximus laoreet.
9 In eget neque at nisl ultricies ornare vitae ut leo. In vulputate nisl ut velit tempus, eget facilisis massa tempus. In sit amet lorem vitae lorem aliquam tempor.
10 Sed in libero ullamcorper risus ultricies venenatis non nec risus. Phasellus bibendum dolor ut nibh dapibus egestas.
11 Vivamus eu enim luctus, blandit augue in, pellentesque nisl. Pellentesque velit eros, malesuada nec varius ac, dictum eget ligula. Ut sed ipsum pharetra...
12
13 Parabens! Codigo secreto: nXzDkn

PROBLEMAS 2 SAÍDA CONSOLE DE DEPURAÇÃO TERMINAL PORTAS
(env) PS D:\Projeto_Python> python .\teste_weak.py
(env) PS D:\Projeto_Python> python .\teste_weak.py
(env) PS D:\Projeto_Python> python .\teste_weak.py
(env) PS D:\Projeto_Python> python .\teste_weak.py
(env) PS D:\Projeto_Python> python .\teste_weak.py
Iniciando ataque de força bruta...
Tentativas até agora: 43,908,363 | Última chave testada: b'SecurityAES28oJy'
Tentativas até agora: 50,134,452 | Última chave testada: b'SecurityAES3ywql'

Possível correspondência encontrada!
Chave: b'SecurityAES3LUbU'
Primeiros 100 bytes do texto decifrado: b'Cras scelerisque pellentesque lectus, quis varius risus varius non. Fusce eu augue at ante bibendum '

Tempo gasto: 510.86 segundos
Chaves testadas: 53324077
Taxa: 104381.21 chaves/seg
None
(env) PS D:\Projeto_Python>
```