


Universidade de Brasília – Departamento de Ciências da Computação			
Disciplina: Segurança Computacional			
Professor(a): Lorena de Souza Bezerra Borges			
Aluno(a): Ayrla Danielly Nascimento Costa			
Matrícula: 190025069	Turma: 2	Data: 20/12/2024	

## RELATÓRIO DO TRABALHO 1 DE SEGURANÇA COMPUTACIONAL

### • Introdução:

Entre os algoritmos de criptografia simétrica, o DES (*Data Encryption Standard*), desenvolvido na década de 1970 pela IBM (Katz & Lindell, 2021), desempenha um papel histórico importante. O S-DES (*Simplified Data Encryption Standard*) é uma versão educacional reduzida do DES, projetada para facilitar a compreensão dos conceitos fundamentais da criptografia de bloco. Este trabalho apresenta a implementação do S-DES em Python, com o objetivo de encriptar e decriptar dados utilizando uma chave de 10 bits e um bloco de dados de 8 bits, conforme descrito nos parâmetros fornecidos.

### • Desenvolvimento:

O algoritmo S-DES opera em blocos de 8 bits, usando uma chave de 10 bits para gerar duas chaves subjacentes de 8 bits através de permutações e rotações, que são invertidas na decritação.

A implementação foi estruturada em funções que realizam as transformações necessárias, como permutações, expansão, substituição e XOR. A seguir, detalham-se as etapas principais do algoritmo e as respectivas funções implementadas:

## Geração de chaves subjacentes

A geração das chaves subjacentes é realizada em três etapas:

1. **Permutação P10**: A chave de 10 bits é reorganizada conforme uma tabela predefinida.
2. **Rotações**: São realizadas rotações à esquerda em duas metades da chave.
3. **Permutação P8**: Aplica-se uma segunda reorganização para gerar chaves subjacentes de 8 bits.

## Permutações

As permutações iniciais (IP), finais ( $IP^{-1}$ ) e intermediárias são responsáveis por reorganizar os bits do texto de entrada. São implementadas usando tabelas que especificam a nova posição de cada bit.

## Rede de Feistel

O processo de encriptação e decríptação utiliza a estrutura de Feistel, dividindo o texto em duas partes de 4 bits:

- **Função F**: Expande e permuta uma metade do texto, realiza uma operação XOR com a chave subjacente e aplica as S-Boxes para substituição de bits. Finalmente, reorganiza o resultado usando a permutação P4.
- **Troca de Partes**: Após a função F, as duas partes são trocadas.

## Substituição (S-Boxes)

As S-Boxes S0 e S1 substituem blocos de 4 bits por novos blocos de 2 bits, com base em tabelas predefinidas que mapeiam combinações de linha e coluna.

## Funções Implementadas

- **criptografar\_SDES**: Realiza a encriptação de um texto plano utilizando as chaves subjacentes geradas e aplica duas iterações da rede de Feistel.
- **descriptografar\_SDES**: Efetua a decriptação de um texto cifrado, utilizando as chaves subjacentes na ordem inversa e duas iterações da rede de Feistel.
- **gerar\_subchave**: Gera duas chaves subjacentes de 8 bits a partir de uma chave de 10 bits, utilizando permutações e rotações.
- **permutacao\_inicial e permutacao\_final**: Realizam a reorganização dos bits do texto de entrada e saída, respectivamente, conforme tabelas predefinidas.
- **expandir\_permutacao**: Expande 4 bits para 8 bits utilizando uma tabela de expansão.
- **permutacao\_10 e permutacao\_8**: Reorganizam os bits da chave de entrada em diferentes etapas do processo de geração de chaves subjacentes.
- **permutacao\_4**: Reorganiza os bits após as substituições nas S-Boxes.
- **sbox\_0 e sbox\_1**: Realizam substituições de blocos de 4 bits por 2 bits, usando tabelas predefinidas.
- **feistel**: Implementa a estrutura de Feistel, incluindo a aplicação da função F e a troca de metades.
- **F**: Aplica a expansão, o XOR com a chave subjacente, as substituições pelas S-Boxes e a permutação P4.
- **xor**: Realiza a operação XOR bit a bit entre duas sequências binárias.
- **pad**: Preenche uma string binária com zeros à esquerda até atingir o tamanho especificado.

## Implementação e Execução

O código foi implementado em Python e respeita as restrições do trabalho, evitando o uso de bibliotecas externas. O bloco de dados fornecido ('11010111') foi

submetido ao algoritmo usando a chave ('1010000010'). O software produz os seguintes resultados:

- Texto Plano: 11010111
- Texto Cifrado: (resultado gerado pelo código)
- Texto Decifrado: 11010111

### **Repositório de acesso**

O link para o repositório no GitHub, contendo o código é:  
<https://github.com/AyrlaCosta/SegurancaComputacional2024.2>.

- **Conclusão:**

O trabalho permitiu a compreensão detalhada dos componentes de um algoritmo de criptografia de bloco, como permutações, substituições e a estrutura de Feistel. A implementação do S-DES em Python proporcionou um exemplo funcional e didático do processo de encriptação e decifração. Além disso, contribuiu para o desenvolvimento de habilidades em implementação de algoritmos criptográficos.

- **Referências:**

KATZ, J.; LINDELL, Y. *Introduction to modern cryptography*. Boca Raton, FL: Crc Press, 2021. P. 212-223

STALLINGS, W. *Cryptography and network security: principles and practice*. 7. ed. Boston: Pearson Education, 2017. p. 714–717