



**INSTITUTO POLITÉCNICO DE BEJA**  
Escola Superior de Tecnologia e Gestão  
Mestrado em Engenharia de Segurança Informática  
Criptografia e Criptanálise Aplicadas

# Desenvolvimento de Aplicação de Cifra Simétrica

Rafael Conceição Narciso - 24473  
Hugo Diogo - 18803



Beja, dezembro de 2025

**INSTITUTO POLITÉCNICO DE BEJA**  
**Escola Superior de Tecnologia e Gestão**  
**Mestrado em Engenharia de Segurança Informática**

# **Desenvolvimento de Aplicação de Cifra Simétrica**

**Rafael Conceição Narciso - 24473**  
**Hugo Diogo - 18803**

Orientador: Prof. Rui Miguel Silva

Beja, dezembro de 2025

## ***Resumo***

Este relatório descreve o desenvolvimento de uma aplicação gráfica para a cifragem e decifragem de ficheiros, implementada na linguagem Python. O projeto visa consolidar conhecimentos de criptografia simétrica através da implementação manual de cifras clássicas (Vigenère e Playfair) e da integração segura de bibliotecas para cifras modernas (AES e DES). São abordadas decisões de arquitetura, modos de operação (CBC) e considerações de segurança.

**Palavras-chave:** Python, Criptografia Simétrica, AES, DES, Vigenère, Playfair.

## ***Abstract***

This report details the development of a graphical application for file encryption and decryption, implemented in Python. The project aims to consolidate symmetric cryptography knowledge through the manual implementation of classical ciphers (Vigenère and Playfair) and the secure integration of libraries for modern ciphers (AES and DES). Architectural decisions, modes of operation (CBC), and security considerations are discussed.

**Keywords:** Python, Symmetric Cryptography, AES, DES, Vigenère, Playfair.

# Índice

<b>1</b>	<b>Introdução</b>	<b>1</b>
<b>2</b>	<b>Enquadramento Teórico</b>	<b>1</b>
2.1	Cifras Clássicas . . . . .	1
2.2	Cifras Modernas . . . . .	1
<b>3</b>	<b>Desenvolvimento da Aplicação</b>	<b>1</b>
3.1	Interface Gráfica (GUI) . . . . .	1
3.2	Implementação: AES e DES . . . . .	1
3.3	Implementação: Vigenère . . . . .	2
3.4	Implementação: Playfair . . . . .	2
<b>4</b>	<b>Análise de Segurança e Decisões de Projeto</b>	<b>2</b>
<b>5</b>	<b>Testes e Resultados</b>	<b>2</b>
<b>6</b>	<b>Conclusão</b>	<b>2</b>

# Índice de Figuras

1	Interface principal da aplicação desenvolvida. . . . .	1
2	Exemplo de operação com AES. . . . .	2

# 1 Introdução

A segurança da informação depende, em grande medida, da robustez dos algoritmos criptográficos utilizados para proteger a confidencialidade dos dados. Este projeto, desenvolvido no âmbito da unidade curricular de Criptografia e Criptanálise Aplicadas, consiste na criação de uma ferramenta em Python capaz de cifrar e decifrar ficheiros utilizando quatro algoritmos distintos: dois clássicos (Vigenère e Playfair) e dois modernos (DES e AES).

A aplicação fornece uma Interface Gráfica (GUI) para facilitar a interação do utilizador, abstraindo a complexidade das operações matemáticas e binárias subjacentes.

## 2 Enquadramento Teórico

### 2.1 Cifras Clássicas

As cifras clássicas implementadas operam ao nível do carácter e pertencem à era pré-computacional.

- **Vigenère:** Uma cifra polialfabética que utiliza uma chave e uma tabela (*Tabula Recta*) para substituir caracteres, dificultando a análise de frequências simples.
- **Playfair:** Uma cifra de substituição poligráfica que opera sobre digramas (pares de letras) utilizando uma matriz  $5 \times 5$ .

### 2.2 Cifras Modernas

As cifras modernas operam sobre blocos de bits e são desenhadas para resistir a ataques computacionais.

- **DES (Data Encryption Standard):** Algoritmo de bloco de 64 bits baseado numa Rede de Feistel. Utiliza uma chave de 56 bits.
- **AES (Advanced Encryption Standard):** Sucessor do DES, operando com blocos de 128 bits e chaves de 128, 192 ou 256 bits, baseado numa rede de substituição-permutação.

## 3 Desenvolvimento da Aplicação

A aplicação foi desenvolvida em **Python** devido à sua versatilidade na manipulação de *byte streams*. A interface gráfica foi construída com a biblioteca **Tkinter**, seguindo uma arquitetura orientada a eventos.

### 3.1 Interface Gráfica (GUI)

A GUI permite ao utilizador selecionar o algoritmo, carregar os ficheiros de chave/tabela e indicar os ficheiros de entrada e saída.

Figura 1: Interface principal da aplicação desenvolvida.

### 3.2 Implementação: AES e DES

Para as cifras modernas, utilizou-se a biblioteca **PyCryptodome**. Ao contrário de implementações triviais que utilizam o modo ECB (Electronic Codebook), optou-se pelo modo **CBC (Cipher Block Chaining)** para garantir a segurança semântica.

**Detalhes de Implementação:**

- **Padding:** Foi aplicado *padding* PKCS7 para ajustar o tamanho dos dados ao tamanho do bloco (8 bytes para DES, 16 para AES).
- **Gestão de IV:** O Vetor de Inicialização (IV) é gerado aleatoriamente e concatenado no início do ficheiro cifrado.

```

1 def encrypt_file(self, data):
2     cipher = AES.new(self.key, AES.MODE_CBC)
3     ct_bytes = cipher.encrypt(pad(data, AES.block_size))
4     # Retorna o IV (16 bytes) + Texto Cifrado
5     return cipher.iv + ct_bytes

```

Listing 1: Cifragem AES com concatenação de IV

### 3.3 Implementação: Vigenère

A cifra de Vigenère foi implementada de raiz. A função `_extend_key` garante que a chave cobre todo o texto. A operação matemática é realizada módulo 26.

```

1 row = ord(key[i]) - ord('A')
2 col = ord(char) - ord('A')
3 encrypted_char = self.table[row][col]

```

Listing 2: Lógica aritmética do Vigenère

### 3.4 Implementação: Playfair

A cifra Playfair envolveu a criação dinâmica da matriz  $5 \times 5$ , fundindo 'I' e 'J'. Foi implementado um pré-processamento para inserir o caráter 'X' entre letras repetidas num digrama.

## 4 Análise de Segurança e Decisões de Projeto

Apesar da funcionalidade correta, identificam-se os seguintes aspectos críticos de segurança:

1. **Gestão de Chaves:** As chaves são lidas de ficheiros de texto simples. Em produção, dever-se-ia utilizar um KMS (Key Management System).
2. **Integridade:** O modo CBC garante confidencialidade, mas não integridade. A aplicação é vulnerável a ataques de modificação de bits ("bit-flipping"), pois não implementa HMAC para autenticação da mensagem.
3. **Memória:** Python não permite limpeza segura de memória, deixando chaves expostas em RAM durante a execução.

## 5 Testes e Resultados

Foram realizados testes de cifra e decifra garantindo que o ficheiro decifrado é binariamente idêntico ao original.

Figura 2: Exemplo de operação com AES.

## 6 Conclusão

O desenvolvimento deste projeto permitiu consolidar a distinção prática entre cifras de fluxo de texto (Clássicas) e cifras de bloco binário (Modernas). A implementação do modo CBC e a gestão manual de tabelas no Playfair foram os principais desafios superados, resultando numa ferramenta funcional e didática.