



Universidade Federal do Piauí  
Departamento de Ciência da Computação

Implementação: Envelope Digital

Claudiney Ryan da Silva  
Ellem Almeida Amorim  
Vinícius Alves de Moura

Disciplina: Segurança em Sistemas  
Professor: CARLOS ANDRE BATISTA DE CARVALHO  
Data: 11 de julho de 2023

# Sumário

<b>1</b>	<b>Introdução</b>	<b>2</b>
<b>2</b>	<b>Funcionamento</b>	<b>2</b>
2.1	Criar um par de chaves RSA . . . . .	2
2.2	Criar um envelope . . . . .	2
2.3	Abrir um envelope . . . . .	3
2.4	Sair . . . . .	3
<b>3</b>	<b>Implementação RSA</b>	<b>3</b>
<b>4</b>	<b>Implementação EAS</b>	<b>3</b>
<b>5</b>	<b>Implementação DES</b>	<b>4</b>
<b>6</b>	<b>Implementação Envelope</b>	<b>4</b>
<b>7</b>	<b>Intalação das Bibliotecas</b>	<b>5</b>
7.1	Bibliotecas Necessarias . . . . .	5
7.2	Rodando o Codigo . . . . .	6
7.3	Ambiente Virtual . . . . .	6
<b>8</b>	<b>Testes e Resultados</b>	<b>7</b>
8.1	Criar um par de chaves RSA . . . . .	7
8.2	Criar um envelope . . . . .	7
8.3	Abrir um envelope . . . . .	8
8.4	Visualizar arquivos criptografados . . . . .	9

# 1 Introdução

Este trabalho visa a implementação de um "Envelope Digital" como parte da disciplina de Segurança em Sistemas Computacionais.

As principais funções implementadas são a criação de um envelope e a abertura de um envelope assinado. Na criação do envelope, são utilizados três elementos de entrada: o arquivo em texto claro, o arquivo da chave pública RSA do destinatário e o algoritmo simétrico escolhido (AES, DES ou RC4). O processamento consiste em gerar uma chave simétrica temporária/aleatória, cifrar o arquivo em claro com essa chave e, em seguida, cifrar a chave temporária com a chave do destinatário. Como resultado, são gerados dois arquivos: um contendo a chave assinada e outro contendo o arquivo criptografado.

## 2 Funcionamento

O arquivo "main.py" fornece uma interface interativa para o usuário realizar operações relacionadas à criação e abertura de envelopes digitais, utilizando diferentes algoritmos simétricos e chaves RSA. O usuário pode selecionar as opções desejadas no menu e executar as funcionalidades correspondentes.

### 2.1 Criar um par de chaves RSA

O usuário pode informar a pasta onde as chaves serão armazenadas e um prefixo opcional para o nome dos arquivos.

A função `generate_key_pair()` do módulo `src.rsa` é chamada para criar o par de chaves.

O usuário recebe uma mensagem de confirmação e é solicitado a pressionar Enter para voltar ao menu.

### 2.2 Criar um envelope

O usuário deve informar o arquivo em claro, o arquivo da chave pública RSA do destinatário e o algoritmo simétrico desejado (AES, DES, RC4).

A função `create_envelope()` do módulo `src.envelope` é chamada para criar o envelope.

Se ocorrer algum erro, uma mensagem de erro é exibida.

O usuário recebe uma mensagem de confirmação e é solicitado a pressionar Enter para voltar ao menu.

## 2.3 Abrir um envelope

O usuário deve informar o caminho do arquivo criptografado do envelope, o caminho da chave criptografada, o algoritmo de descriptografia desejado (AES, DES, RC4) e o caminho da chave privada RSA.

A função `open_envelope()` do módulo `src.envelope` é chamada para abrir o envelope.

Se ocorrer algum erro, uma mensagem de erro é exibida. O usuário recebe uma mensagem de confirmação e é solicitado a pressionar Enter para voltar ao menu.

## 2.4 Sair

O programa é finalizado e a função `menu()` retorna.

# 3 Implementação RSA

O código é responsável por gerar um par de chaves RSA, carregar chaves públicas e privadas, além de criptografar e descriptografar dados usando essas chaves.

A função `generate_key_pair` gera um par de chaves RSA com tamanho de chave de 2048 bits e expoente público de 65537. Em seguida, as chaves são serializadas no formato PEM e salvas em arquivos.

A função `load_public_key` carrega a chave pública RSA de um arquivo PEM especificado pelo caminho fornecido. A chave pública é retornada como um objeto.

A função `load_private_key` carrega a chave privada RSA de um arquivo PEM especificado pelo caminho fornecido. A chave privada é retornada como um objeto.

A função `encrypt` criptografa os dados usando a chave pública RSA e o esquema de padding OAEP (Optimal Asymmetric Encryption Padding). O arquivo em claro é passado como entrada e o arquivo cifrado é retornado.

A função `decrypt` descriptografa os dados usando a chave privada RSA e o mesmo esquema de padding OAEP. O arquivo cifrado é passado como entrada e o arquivo em claro é retornado.

Por fim, é chamada a função `generate_key_pair` para gerar um novo par de chaves com um nome específico.

# 4 Implementação EAS

O código implementa funções para criptografar e descriptografar utilizando o algoritmo AES (Advanced Encryption Standard) no modo de operação CBC (Cipher

Block Chaining).

A função `encrypt(key, plaintext)` recebe uma chave e um texto plano como entrada. Ela gera um vetor de inicialização (IV) aleatório, cria um objeto `Cipher` com a chave e o modo CBC, realiza o padding do texto plano usando o esquema PKCS7, cria um objeto de criptografia e criptografa o texto plano com o padding. O resultado é o texto cifrado, que consiste na concatenação do IV e do texto cifrado.

A função `decrypt(key, ciphertext)` recebe uma chave e um texto cifrado como entrada. Ela extrai o IV do início do texto cifrado, cria um objeto `Cipher` com a chave e o modo CBC, cria um objeto de descriptografia e descriptografa o texto cifrado. Em seguida, remove o padding do texto plano descriptografado. O resultado é o texto plano original.

A função `generate_key(size)` gera uma chave aleatória de tamanho especificado em bytes.

## 5 Implementação DES

O código implementa funções para criptografar e descriptografar utilizando o algoritmo DES (Data Encryption Standard) no modo de operação CBC (Cipher Block Chaining).

A função `encrypt(key, plaintext)` recebe uma chave e um texto plano como entrada. Ela gera um vetor de inicialização (IV) aleatório, cria um objeto `Cipher` com a chave e o modo CBC usando o algoritmo DES, realiza o padding do texto plano usando o esquema PKCS7, cria um objeto de criptografia e criptografa o texto plano com o padding. O resultado é o texto cifrado, que consiste na concatenação do IV e do texto cifrado.

A função `decrypt(key, ciphertext)` recebe uma chave e um texto cifrado como entrada. Ela extrai o IV do início do texto cifrado, cria um objeto `Cipher` com a chave e o modo CBC usando o algoritmo DES, cria um objeto de descriptografia e descriptografa o texto cifrado. Em seguida, remove o padding do texto plano descriptografado. O resultado é o texto plano original.

A função `generate_key(size)` gera uma chave aleatória de tamanho especificado em bytes.

## 6 Implementação Envelope

O código é responsável por criar e abrir um envelope digital para criptografia e descriptografia de arquivos usando algoritmos simétricos (AES, DES, RC4) e crip-

tografia assimétrica (RSA).

As funções `create_envelope` e `open_envelope` são responsáveis por criar e abrir o envelope digital, respectivamente. A função `create_envelope` recebe o caminho do arquivo em claro, o caminho da chave pública do destinatário e o algoritmo simétrico a ser utilizado. A função `open_envelope` recebe o caminho do arquivo criptografado, o caminho da chave simétrica criptografada, o caminho da chave privada do destinatário e o algoritmo simétrico utilizado.

Dentro das funções `create_envelope` e `open_envelope`, o código realiza as etapas necessárias para criar e abrir o envelope digital. Isso inclui carregar as chaves, verificar a validade da chave pública, gerar uma chave simétrica temporária/aleatória, criptografar/descriptografar o arquivo em claro usando o algoritmo simétrico, criptografar/descriptografar a chave simétrica usando a chave pública/privada, e salvar os arquivos resultantes.

O resultado da execução do código é a impressão de mensagens indicando se a operação foi concluída com sucesso ou se ocorreu algum erro.

## 7 Instalação das Bibliotecas

Antes de adentrar na sessão de resultados e rodar o código iremos demonstrar como preparar o ambiente.

### 7.1 Bibliotecas Necessárias

Aqui antes de baixar qualquer biblioteca necessária, é preciso atualizar a biblioteca `pip` do `python` ou baixar a mesma, para dar continuidade.

```
1 python -m pip install --upgrade pip
```

Listing 1: Atualizando Biblioteca `pip`

Agora pode se utilizar essa linha de código no prompt podendo assim instalar duas bibliotecas que serão totalmente necessárias para poder rodar o código.

```
1 pip install -r requirements.txt
```

Listing 2: Instalando os Requerimentos

## 7.2 Rodando o Código

Ao utilizar essa linha de comando é possível rodar a interface que será explicada na próxima sessão para poder efetuar o envelopamento do arquivo em claro.

```
1 python main.py
```

Listing 3: Compilando main.py

## 7.3 Ambiente Virtual

Se ao tentar efetuar a instalação das bibliotecas, e mesmo assim as importações apresentarem erro ao dizer que o modulo não está sendo encontrado, é possível utilizar de um ambiente virtual para isolar e gerenciar de forma independente as dependências e configurações de um projeto específico.

```
1 python -m venv .venv
```

Listing 4: Criando Ambiente Virtual no Windows

```
1 .venv\Scripts\activate
```

Listing 5: Ativando Ambiente

O comando "Set-ExecutionPolicy -ExecutionPolicy Bypass -Scope Process -Force" é usado no Windows para alterar a política de execução do PowerShell.

A política de execução do PowerShell é uma configuração de segurança que determina quais scripts podem ser executados no sistema. Por padrão, o PowerShell tem uma política de execução restritiva que impede a execução de scripts não assinados.

```
1 Set-ExecutionPolicy -ExecutionPolicy Bypass -Scope Process -Force
```

Listing 6: Removendo algumas restrições

```
1 source .venv/bin/activate
```

Listing 7: Criando um ambiente no Linux/Mac

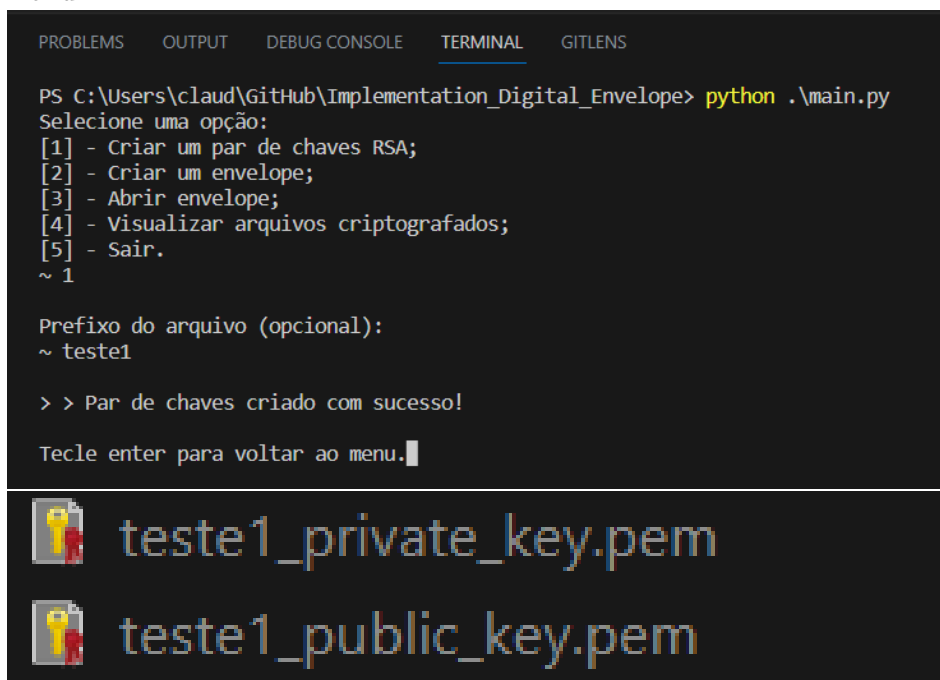
Após criar esse ambiente e ativa-lo, já é possível efetuar a instalação das bibliotecas necessárias ditas anteriormente.

## 8 Testes e Resultados

Daqui em diante será mostrado alguns exemplos com o funcionamento do programa, mostrando etapa por etapa de como ocorre a criptografia do arquivo em claro e das chaves simétricas. Será usado um arquivo teste1.txt para ser criptografado com o algoritmo AES, e também é valido dizer que o programa consegue criptografar qualquer extensão (.png, .jpeg, .pdf, etc.)

### 8.1 Criar um par de chaves RSA

Selecionando a opção 1 solicita ao usuário um prefixo para o arquivo de chave. Chama a função `rsa.generate_key_pair()` do módulo `src.rsa` para gerar um par de chaves RSA. Se ocorrer algum erro durante a geração das chaves, o erro é capturado e exibido na tela. Após a execução, aguarda o usuário pressionar "Enter" para voltar ao menu.



```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL GITLENS

PS C:\Users\claud\GitHub\Implementation_Digital_Envelope> python .\main.py
Selecione uma opção:
[1] - Criar um par de chaves RSA;
[2] - Criar um envelope;
[3] - Abrir envelope;
[4] - Visualizar arquivos criptografados;
[5] - Sair.
~ 1

Prefixo do arquivo (opcional):
~ teste1

> > Par de chaves criado com sucesso!

Tecle enter para voltar ao menu.

🔑 teste1_private_key.pem
🔑 teste1_public_key.pem
```

### 8.2 Criar um envelope

Na opção 2 solicita ao usuário o caminho para o arquivo que deseja criptografar. Solicita ao usuário o caminho para o arquivo da chave pública (.pem). Solicita ao usuário o algoritmo simétrico de criptografia (AES, DES ou RC4). Chama a função `envelope.create_envelope()` do módulo `src.envelope` para criar um envelope com o arquivo criptografado. Se ocorrer algum erro durante a criação do envelope, o erro é capturado e exibido na tela. Após a execução, aguarda o usuário pressionar



"Enter" para voltar ao menu.

```
Selecione uma opção:
[1] - Criar um par de chaves RSA;
[2] - Criar um envelope;
[3] - Abrir envelope;
[4] - Visualizar arquivos criptografados;
[5] - Sair.
~ 2



Caminho para arquivo que deseja criptografar:
~ teste1.txt

Caminho para a chave pública (.pem):
~ teste1_public_key.pem

Algoritmo simétrico: [AES, DES, RC4]:
> AES

> > Envelope criado com sucesso!

Tecle enter para voltar ao menu.

 encrypted_simetric_key.key M
 encrypted_teste1.txt U
```

### 8.3 Abrir um envelope

Na opção 3 solicita ao usuário o caminho para o arquivo criptografado. Solicita ao usuário o caminho para o arquivo da chave simétrica criptografada (.key). Solicita ao usuário o caminho para o arquivo da chave privada (.pem). Solicita ao usuário o algoritmo simétrico de descriptografia (AES, DES ou RC4). Chama a função `envelope.open_envelope()` do módulo `src.envelope` para abrir o envelope e obter o `arquivo_original` descriptografado. Se ocorrer algum erro durante a abertura do envelope, o erro é capturado e exibido na tela. Após a execução, aguarda o usuário pressionar "Enter" para voltar ao menu.

```

Selecione uma opção:
[1] - Criar um par de chaves RSA;
[2] - Criar um envelope;
[3] - Abrir envelope;
[4] - Visualizar arquivos criptografados;
[5] - Sair.
~ 3

Caminho para o arquivo criptografado:
~ encrypted_teste1.txt

Caminho para a chave simétrica criptografada (.key):
~ encrypted_simetric_key.key

Caminho para a chave privada (.pem):
~ teste1_private_key.pem

Algoritmo simétrico: [AES, DES, RC4]:
> AES

> > Envelope aberto com sucesso!

Tecle enter para voltar ao menu.

```



decrypted\_encrypted\_teste1.txt



## 8.4 Visualizar arquivos criptografados

Escolhendo a opção 4 solicita ao usuário o caminho para o arquivo criptografado que deseja visualizar. Chama a função `utils.b64_encode()` do módulo `src.utils` para codificar o arquivo em Base64 e exibir na tela. Após a execução, aguarda o usuário pressionar "Enter" para voltar ao menu.

```

Selecione uma opção:
[1] - Criar um par de chaves RSA;
[2] - Criar um envelope;
[3] - Abrir envelope;
[4] - Visualizar arquivos criptografados;
[5] - Sair.
~ 4

Caminho para o arquivo criptografado que deseja visualizar:
~ encrypted_teste1.txt
b'c9e0aGx+7tzWLQYDJATBX+0EfQr4nMRSSnRKMQUIo0Wp7mJ/rk5UgeYYk4kJsUCyDeNp3pTeb
+88JuntPFVQGouhZRM7t6+0SxhZ0Jv2QAwM3Gd1Fqgfw/xlp4YJreU1w5UkOclal5vlbfvCbml
jEWxtJ4mXTJD0w9J3YyF+0iCvtOVURqmk5GAYiG1RwRowNdMIpg8FMh6sC1wYohy6KuL4LMUuk
1ifCtqm3MMEDpr1biiEVvgE5Qil+ePE3WtpeCIOgJY+9TSULDrBgOXu4haaw57wAhAjzA9Ssp8
qvehNw5Bx1ez+FdH9CYJNv4L91GsissDF5Z5FMuXgiQkw8MuSLf3FSwBXoHB18toqyRrlyU1rx
GB1HERSY='

Tecle enter para voltar ao menu.

```

```
Selecione uma opção:
[1] - Criar um par de chaves RSA;
[2] - Criar um envelope;
[3] - Abrir envelope;
[4] - Visualizar arquivos criptografados;
[5] - Sair.
~ 4

Caminho para o arquivo criptografado que deseja visualizar:
~ encrypted_simetric_key.key
b'UE9ZZU1yUHLqcEY0duLoa0NyMTdWUHRraEhiQmNaUzJjVmRrSVU1awZnbnljQTV5Z
LN1dsWFVMQVB5R0hXQ3dTNHhiTWRFMXpIRnVhaVk2U01iWG9SbVJqL1FUK1owamR2QU
cnpvem1RT0xSRHA4NytEWkFFcnVKQ203dWowRHovMC9LYlNMTGxGK2pSdnM1RnRRdVp
Tecle enter para voltar ao menu.
```

Sobre as chaves RSA, ao abrir os arquivos tanto da chave publica quanto da chave privada, é possível visualizar o conjunto de caracteres usados. O arquivo em claro criptografado também é possível de ser visualizado ao ser aberto, igualmente com o arquivo descriptografado.