UNIVERSIDADE FEDERAL DO PIAUÍ – UFPI
CENTRO DE CIÊNCIAS DA NATUREZA – CCN
DEPARTAMENTO DE COMPUTAÇÃO
CURSO DE BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO
Disciplina: Segurança em Sistemas Computacionais
Professor da disciplina: Carlos André Batista de Carvalho
Aluno: Lemuel Cavalcante Lopes **Matrícula: 20209063994**

Trabalho 01 – Implementação: Sistema de Gerenciamento de Chaves Públicas e Criptografia

Relatório de Implementação e descrição de instruções de utilização do algoritmo RSA

Linguagem utilizada: "Python"
Source Code "gerar_chaves.py"
Par de chaves publicas e privadas de extensao: ".pem"
Arquivo a ser cifrado: "documento.txt"

1. **Introdução**

Este programa implementa funcionalidades de criptografia e descriptografia de arquivos usando o algoritmo RSA (Rivest-Shamir-Adleman). Ele permite gerar, salvar e carregar chaves públicas e privadas, além de criptografar e descriptografar dados usando essas chaves.

2. **Instruções para Execução**

Instalação de Bibliotecas: Certifique-se de que as bibliotecas `"cryptography"` e `"os"` estejam instaladas. Você pode instalar as dependências executando o seguinte comando no terminal:

```
"pip install cryptography"
```

3. **Geração de Chaves:**

Para gerar um novo par de chaves, execute o programa e chame a função:

```
"gerar_par_chaves()"
```

As chaves serão geradas com um tamanho de 2048 bits e um expoente público de 65537.

4. **Salvamento e Carregamento de Chaves:**

Use as funções "salvar_chave_privada" e "salvar_chave_publica" para salvar chaves em arquivos.
Use as funções "carregar_chave_privada" e "carregar_chave_publica" para carregar chaves de arquivos.

5. **Gerenciamento de Chaves Armazenadas:**

Utilize as funções "listar_arquivos_diretorio", "pesquisar_arquivo" e "apagar_arquivo" para listar, pesquisar e apagar arquivos de chaves armazenadas, respectivamente.

6. **Proteção de Chave Privada com Senha:**

Ao salvar uma chave privada em um arquivo, você pode protegê-la com uma senha fornecendo um parâmetro senha para a função "salvar_chave_privada".

7. **Criptografia e Descriptografia de Dados:**

Para criptografar dados, use a função "criptografar_arquivo" fornecendo o caminho do arquivo de chave pública, o arquivo de entrada e o arquivo de saída.

Para descriptografar dados, use a função "descriptografar_arquivo" fornecendo o caminho do arquivo de chave privada, o arquivo cifrado de entrada e o arquivo de saída.

Os dados são criptografados e descriptografados usando o algoritmo RSA, com padding OAEP (Optimal Asymmetric Encryption Padding) para garantir segurança e integridade.

8. **Explicações sobre as Funções Utilizadas**

- cryptography.hazmat.primitives.asymmetric.rsa.generate_private_key:

Esta função é usada para gerar uma chave privada RSA.

Parâmetros:

"public_exponent:" O expoente público. O padrão é 65537, que é comumente usado.

"key_size:" O tamanho da chave em bits. O padrão é 2048 bits.

"backend:" O backend de criptografia a ser usado. O padrão é o backend padrão da biblioteca cryptography.

- `cryptography.hazmat.primitives.asymmetric.rsa.generate_private_key:`

Esta função é usada para gerar uma chave pública correspondente a uma chave privada RSA.

Parâmetros:

"`private_key`": A chave privada da qual a chave pública será derivada.

- `cryptography.hazmat.primitives.asymmetric.rsa.generate_private_key:`

Esta função é usada para gerar uma chave privada RSA em formato PEM.

Parâmetros:

"`encoding:`" O formato de codificação do arquivo PEM.
"`format:`" O formato da chave privada.
"`encryption_algorithm`": O algoritmo de criptografia a ser usado para proteger a chave privada com uma senha.

`cryptography.hazmat.primitives.asymmetric.rsa.generate_private_key:`

Esta função é usada para carregar uma chave privada RSA de um arquivo PEM.

Parâmetros:

"`password`": A senha para descriptografar a chave privada, se protegida.

- `cryptography.hazmat.primitives.asymmetric.rsa.generate_private_key:`

  Esta função é usada para salvar uma chave pública RSA em formato PEM.

  Parâmetros:

"`encoding:`" O formato de codificação do arquivo PEM.
"`format:`" O formato da chave pública.

- `cryptography.hazmat.primitives.asymmetric.rsa.generate_private_key:`

Esta função é usada para carregar uma chave pública RSA de um arquivo PEM.