Universidade de Brasília

Curso de Ciências da Computação

CIC0201 - Segurança Computacional - 2024/1

Algoritmos de Criptografia: AES e RSA

Este trabalho tem como objetivo explorar os algoritmos de implementação de criptografia AES (Advanced Encryption Standard) e RSA (Rivest-Shamir-Adleman), destacando suas características, dificuldades e segurança. A criptografia é fundamental na proteção de informações sensíveis, e esses dois algoritmos são amplamente utilizados em diversas aplicações, desde a proteção de dados pessoais até a segurança de transações bancárias e comunicações digitais.

Aluno: Matheus Azevedo Oliveira - 17/0110940

Professor: João Gondim

Brasília - 2024

Introdução

A criptografia é um elemento essencial na segurança da informação, garantindo que dados sensíveis sejam protegidos contra acesso não autorizado. Entre os muitos algoritmos de criptografia disponíveis, focaremos no AES (Advanced Encryption Standard) e o RSA (Rivest-Shamir-Adleman) para a finalização deste relatório. O AES é um algoritmo de criptografia simétrica, amplamente adotado por sua velocidade e eficiência em proteger dados. Por outro lado, o RSA é um algoritmo de criptografia assimétrica, conhecido por sua capacidade de garantir a segurança através de chaves públicas e privadas. Este relatório explora a implementação, características e desafios de ambos os algoritmos.

AES (Advanced Encryption Standard)

1. Implementação de Código:

O AES é implementado utilizando um processo de substituição-permutação, onde os dados são organizados em blocos de 128 bits e passam por uma série de transformações baseadas em chaves criptográficas de 128, 192 ou 256 bits (neste relatório, iremos considerar somente chaves de 128 bits). A implementação pode ser feita em diversas linguagens de programação; por fins de afinidade, irei utilizar Python para essa implementação.

2. Features Implementadas:

- Cifração: Possível cifrar plaintext (string) ou arquivos
- Decifração: Possível decifrar plaintext (string) ou arquivos
- Mudança de modo de cifra: ECB ou CTR
- Escolha de quantidade de rounds para cifrar
- Quando no modo CTR: possível escolher valor inicial do contador e valor de incremento

3. Dificuldades Encontradas:

- Entender como montar as operações matemáticas em código
- Deixar um código performático
- Testar os dados que eram produzidos
- Encontrar amostra de testes

4. Testes Realizados:

- Testes de Performance: Foram realizados testes de performance para medir o tempo de criptografia e decriptografia de diferentes tamanhos de dados.
- Testes de Corretude: Testes para verificar a corretude e acerto dos dados cifrados e decifrados.

5. Resultados de Cifrar/Decifrar uma Selfie:

Ao cifrar uma mesma selfie, utilizando números de rounds distintos, houve somente uma mudança no tempo de processamento das cifras. Em todos os casos, não foi possível renderizar o resultado cifrado, sendo necessário decifrar o resultado para renderizar a imagem novamente. Juntamente com o código fonte da aplicação, deixo os arquivos que constam os resultados das cifras, respectivamente no rounds 1, 5, 9 e 13. Os arquivos estão com o nome na seguinte forma: “out-file{p}.txt”, onde p é a quantidade de rounds que foi utilizada para gerar tal arquivo. Todos estão codificados em Base64.

Além disso, deixo os dados usados para gerar as cifras:
Key: MySecretKey59283
Mode: ECB
Foto: foto-perfil.jpeg

O tempos de processamento de cada cifra ficaram da seguinte forma:

- 1 round: 2.9s
- 5 rounds: 9.3s
- 9 rounds: 15.1s
- 13 rounds: 21.5s

RSA (Rivest-Shamir-Adleman)

1. Implementação de Código:

O RSA utiliza duas chaves diferentes para criptografia e decriptografia, uma chave pública e uma chave privada. A implementação básica envolve a geração de pares de chaves e a utilização de operações matemáticas de exponenciação modular para cifrar e decifrar as mensagens.

2. Features Implementadas:

- Geração de chaves: Possível gerar chaves de tamanho mínimo 1024 bits
- Geração de Cifra: Possível gerar uma cifra
- Geração de Decifrar: Possível decifrar um arquivo cifrado

3. Dificuldades Encontradas:

- Montar o algoritmo para assinatura
- Criar fluxo de conversão de dados (ints para bytes)
- Busca de números primos