Trabalho de Criptografia em Python

Este trabalho tem como objetivo pesquisar, implementar, codificar e verificar algoritmos criptográficos utilizando a linguagem de programação Python. Foram abordadas três frentes principais: criptografia simétrica, criptografia assimétrica e funções hash.

1. Criptografia com Chaves Simétricas (AES-CBC)

Na criptografia simétrica, a mesma chave é usada tanto para criptografar quanto para descriptografar. Neste trabalho foi utilizada a cifra AES no modo CBC com padding PKCS#7.

```
from cryptography.hazmat.primitives.ciphers import Cipher, algorithms, modes
from cryptography.hazmat.primitives import padding
from cryptography.hazmat.backends import default_backend
import os

key = os.urandom(32)
iv = os.urandom(16)
cipher = Cipher(algorithms.AES(key), modes.CBC(iv), backend=default_backend())

padder = padding.PKCS7(128).padder()
plaintext = b"Mensagem secreta"
padded = padder.update(plaintext) + padder.finalize()
encryptor = cipher.encryptor()
ciphertext = encryptor.update(padded) + encryptor.finalize()
```

2. Criptografia com Chaves Assimétricas (RSA-OAEP)

Na criptografia assimétrica, utilizam-se duas chaves: uma pública e outra privada. A implementação a seguir utiliza RSA com OAEP e SHA-256.

```
from cryptography.hazmat.primitives.asymmetric import rsa, padding
from cryptography.hazmat.primitives import hashes

private_key = rsa.generate_private_key(public_exponent=65537, key_size=2048)
public_key = private_key.public_key()

message = b"Mensagem para RSA"
ciphertext = public_key.encrypt(message, padding.OAEP(
    mgf=padding.MGF1(algorithm=hashes.SHA256()),
    algorithm=hashes.SHA256(),
    label=None
))
plaintext = private_key.decrypt(ciphertext, padding.OAEP(
    mgf=padding.MGF1(algorithm=hashes.SHA256()),
    algorithm=hashes.SHA256(),
    label=None
))
```

3. Funções Hash (SHA-256)

As funções hash são algoritmos unidirecionais que geram um resumo fixo de uma entrada. No exemplo a seguir foi utilizada a função SHA-256.

```
import hashlib

message = b"Exemplo de mensagem"
hash_value = hashlib.sha256(message).hexdigest()
print(hash_value)
```

Conclusão

Foram implementados e verificados três tipos de algoritmos criptográficos: AES-CBC (simétrico), RSA-OAEP (assimétrico) e SHA-256 (hash). Cada técnica possui aplicações específicas: AES é eficiente para grandes volumes de dados, RSA é utilizado para troca segura de chaves e assinaturas digitais, e funções hash são fundamentais para integridade e autenticação.