Trabalho de Implementação 2: Gerador e Verificador de Assinaturas RSA

Vinícius Dos Santos Tomé
Segurança Computacional - CIC0201
Universidade de Brasíia - UnB
Matrícula: 232006780

***Resumo*—Este relatório detalha a implementação de uma ferramenta de linha de comando para a geração e verificação de assinaturas digitais utilizando o algoritmo RSA. O projeto aplica conceitos de criptografia assimétrica, funções de hash SHA-3 e o padrão de preenchimento OAEP para garantir a autenticidade e a integridade de ficheiros digitais, cumprindo os requisitos propostos pela disciplina de Segurança Computacional.**

***Index Terms*—Assinatura Digital, RSA, Criptografia Assimétrica, Segurança Computacional.**

I. Introdução

No contexto da Segurança Computacional, a garantia de que a informação digital não foi alterada (integridade) e que provém de uma fonte legítima (autenticidade) é fundamental. As assinaturas digitais surgem como uma das principais ferramentas criptográficas para prover estes serviços de segurança. Este trabalho aborda a implementação prática de um sistema de geração e verificação de assinaturas digitais baseado no algoritmo de chave pública RSA [1].

O objetivo deste projeto é desenvolver uma ferramenta funcional que aplique os conceitos teóricos de criptografia assimétrica para criar um mecanismo robusto de assinatura. Este relatório está estruturado da seguinte forma: a secção II descreve a fundamentação teórica, a secção III detalha o ambiente experimental e os resultados, e a secção IV apresenta as conclusões.

II. Fundamentação Téorica

Para a construção da ferramenta, foram aplicados diversos conceitos criptográficos que, em conjunto, formam um sistema de assinatura digital seguro.

A. Criptografia Assimétrica e o Algoritmo RSA

Diferente da criptografia simétrica, a criptografia assimétrica emprega um par de chaves: uma pública e uma privada. O algoritmo RSA, proposto por Rivest, Shamir e Adleman, é o padrão mais difundido para este fim [1]. A sua segurança reside na dificuldade computacional de fatorar números inteiros muito grandes. O processo envolve a escolha de dois primos grandes, p e q, a partir dos quais se calcula o módulo $n = p \times q$ e a função totiente $\phi(n) = (p-1)(q-1)$.

B. Assinatura Digital

Uma assinatura digital vincula a identidade do signatário a um documento. O processo consiste em calcular um resumo criptográfico (hash) da mensagem e cifrar este hash com a chave privada do signatário. A verificação é feita decifrando a assinatura com a chave pública e comparando o hash resultante com um novo hash da mensagem recebida. Se forem idênticos, a assinatura é válida [2].

C. Funções de Hash e OAEP

Este projeto utiliza o SHA-3, o mais recente padrão do NIST para funções de hash, pela sua robustez [3]. Adicionalmente, para mitigar vulnerabilidades do RSA "textbook", foi implementado o padrão de preenchimento OAEP (Optimal Asymmetric Encryption Padding), que adiciona aleatoriedade à mensagem antes da cifragem, tornando o RSA semanticamente seguro [4].

III. Ambiente Experimental e Análise de Resultados

A. Descrição do Cenário

- **Hardware e Software:** O desenvolvimento foi realizado num computador pessoal com sistema operativo Windows 11. A linguagem utilizada foi Python 3.11, com a biblioteca `pycryptodome` para manipulação de chaves no formato PEM.
- **Topologia de Rede:** Por ser uma aplicação para criptografia de ficheiros locais, não foi utilizada uma topologia de rede.
- **Configurações:** A ferramenta foi desenvolvida como uma interface de linha de comando (CLI) utilizando o módulo `argparse` do Python. Os comandos implementados são `gerar-chaves`, `assinar` e `verificar`.

B. Análise de Resultados

Foi executado um roteiro de testes para validar cada funcionalidade exigida, conforme descrito abaixo.

- **Atividade 1: Geração de Chaves.** O comando `py geradorVerificador.py gerar-chaves` resultou na criação bem-sucedida dos ficheiros `chave_privada.pem` e `chave_publica.pem`.
- **Atividade 2: Assinatura e Verificação bem-sucedida.** Um ficheiro de texto foi assinado e, ao ser verificado, o

sistema retornou "ASSINATURA VÁLIDA", exibindo o conteúdo original corretamente.

- **Atividade 3: Teste de Integridade.** Um único caractere do ficheiro assinado (codificado em Base64) foi alterado. Ao tentar a verificação, o programa retornou "ASSINATURA INVÁLIDA", comprovando a deteção de adulteração.
- **Atividade 4: Teste de Autenticidade.** Um documento assinado com um par de chaves antigo foi testado com um novo par de chaves. A verificação falhou, confirmando que a assinatura está unicamente ligada ao par de chaves original.

Durante a configuração do ambiente, foi necessário resolver um problema de aliases de execução do Python no Windows, utilizando o lançador `py` para garantir a execução com a versão correta do interpretador.

IV. Conclusões

A implementação da ferramenta de assinatura digital RSA foi concluída com sucesso, cumprindo todos os requisitos funcionais e de segurança propostos. O trabalho permitiu a aplicação prática de conceitos teóricos da criptografia assimétrica, resultando numa ferramenta robusta que valida corretamente a integridade e a autenticidade de ficheiros. O projeto reforça a compreensão de como as assinaturas digitais fornecem uma base de confiança essencial para a segurança da informação no mundo digital.

Referências

[1] R. L. Rivest, A. Shamir, and L. Adleman, "A method for obtaining digital signatures and public-key cryptosystems,"*Communications of the ACM*, vol. 21, no. 2, pp. 120–126, 1978.

[2] W. Stallings, *Cryptography and Network Security: Principles and Practice*, 7th ed. Pearson, 2017.

[3] National Institute of Standards and Technology (NIST), "FIPS PUB 202: SHA-3 Standard: Permutation-Based Hash and Extendable-Output Functions,"2015.

[4] M. Moriarty, B. Kaliski, J. Jonsson, A. Rusch, "PKCS #1: RSA Cryptography Specifications Version 2.2,"RFC 8017, 2016.

[5] PyCryptodome Developers, "PyCryptodome Documentation,"[Online]. Available: https://www.pycryptodome.org/en/latest/.