Projeto: Comunicação Segura via TCP com Cifra RSA

Este projeto implementa uma comunicação segura entre um cliente e um servidor utilizando a cifra RSA para criptografia das mensagens transmitidas. O sistema é composto por dois componentes principais:

- **Servidor TCP (Bob)**: Responsável por gerar as chaves RSA, receber mensagens criptografadas do cliente, descriptografar essas mensagens e enviar a resposta em caixa alta.
- **Cliente TCP (Alice)**: Conecta-se ao servidor, recebe a chave pública RSA, criptografa uma mensagem e a envia para o servidor.

Arquivos

- `Simple_tcpServer.py` : Este script implementa o servidor TCP. Ele gera as chaves RSA, realiza o handshake com o cliente, descriptografa a mensagem recebida e responde ao cliente com a mensagem em caixa alta.
- `Simple_tcpClient.py` : Este script implementa o cliente TCP. Ele se conecta ao servidor, realiza o handshake para receber a chave pública RSA, criptografa uma mensagem e a envia ao servidor.

Funcionamento

Cifra RSA

A cifra RSA é um dos algoritmos de criptografia assimétrica mais utilizados no mundo. Diferente dos métodos de criptografia simétrica, onde a mesma chave é usada para criptografar e descriptografar os dados, na criptografia assimétrica RSA, utiliza-se um par de chaves: uma chave pública para criptografar e uma chave privada para descriptografar.

Principais Etapas:

1. **Geração de Chaves**: O RSA gera um par de chaves (pública e privada) a partir de dois grandes números primos. A segurança do RSA depende da dificuldade em fatorar números grandes.
2. **Criptografia**: Uma mensagem (convertida para um número inteiro) é criptografada usando a chave pública com a fórmula `c = m^e mod n`, onde `c` é o texto cifrado, `m` é a mensagem, `e` é o expoente público, e `n` é o produto dos dois primos.
3. **Descriptografia**: O texto cifrado é então convertido de volta para a mensagem original usando a chave privada com a fórmula `m = c^d mod n`, onde `d` é o expoente privado.

Relação entre as Variáveis `d` e `e`:

As variáveis `d` e `e` são cruciais no funcionamento da cifra RSA:

- `e`: Conhecido como o expoente público, é usado na criptografia da mensagem. Ele é escolhido de forma que seja relativamente pequeno e que `gcd(e, ϕ(n)) = 1`, onde `ϕ(n)` é a função totiente de Euler.
- `d`: Conhecido como o expoente privado, é usado na descriptografia da mensagem. Ele é calculado como o inverso modular de `e` em relação a `ϕ(n)`, ou seja, `d` satisfaz a relação `d * e ≡ 1 (mod ϕ(n))`. Essa relação garante que a criptografia e a descriptografia sejam operações inversas.

Essa relação é fundamental para a segurança do RSA, pois a determinação de `d` a partir de `e` sem o conhecimento dos fatores primos de `n` (os números primos usados na geração de `n`) é computacionalmente inviável.

Algoritmo de Miller-Rabin

Para garantir a segurança das chaves RSA, é necessário que os números usados para gerar as chaves sejam primos. O algoritmo de Miller-Rabin é um teste probabilístico de primalidade usado para verificar se um número é provavelmente primo. Ele realiza múltiplas iterações para determinar a probabilidade de que um número seja primo. Este método é altamente confiável e amplamente utilizado na geração de chaves criptográficas.

Processo de Comunicação

1. **Handshake**: A chave pública gerada pelo servidor é enviada ao cliente.
2. **Envio de Mensagem**: O cliente utiliza a chave pública para criptografar uma mensagem e a envia para o servidor.
3. **Descriptografia e Resposta**: O servidor descriptografa a mensagem recebida usando sua chave privada, converte o texto para caixa alta e o retorna ao cliente.

Como Executar

1. Iniciar o Servidor (Bob)

Execute o script `Simple_tcpServer.py` para iniciar o servidor.

```
python3 Simple_tcpServer.py
```

2. Iniciar o Cliente (Alice)

Execute o script `Simple_tcpClient.py` após o servidor estar em execução.

```
python3 Simple_tcpClient.py
```

Funcionamento

1. **Geração de Chaves RSA**: O servidor gera um par de chaves RSA (pública e privada) de 4096 bits.
2. **Handshake**: A chave pública é enviada ao cliente.
3. **Envio de Mensagem**: O cliente criptografa uma mensagem utilizando a chave pública do servidor e a envia.
4. **Descriptografia e Resposta**: O servidor descriptografa a mensagem, a converte para caixa alta e envia de volta ao cliente.

Autores

- Abrão Asterio Junior
- Alexandre Bezerra de Andrade
- Daniel Santos de Sousa
- Francisco Tommasi Silveira