Logotipo

Descrição gerada automaticamente

**Sistemas Distribuídos**

**Aluno: Marcos Melo**

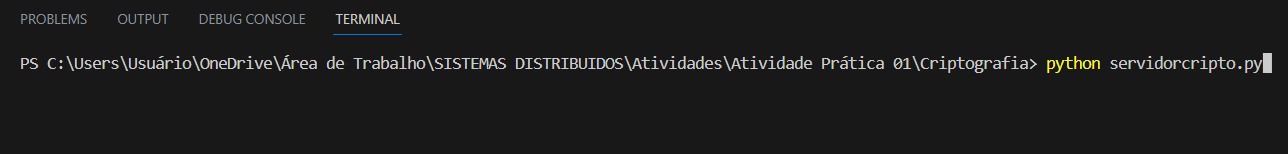
**Atividade Prática 01**

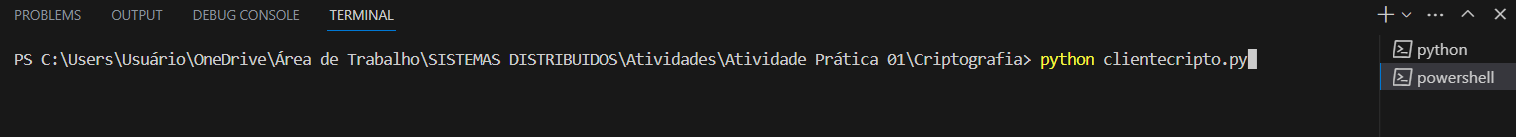
**Os códigos do servidor e cliente vão estar dentro da pasta “criptografia”**

**Extensão necessária para a criptografia:**

**pip install pycryptodome**

Para rodar o código vamos abrir dois terminais pelo próprio vscode

1 – Começamos iniciando o servidor através da linha: python servidorcripto.py

2 – Depois iniciamos o cliente através: python clientecripto.py

O código consiste em um cliente e um servidor que se comunicam por meio de sockets TCP/IP. O cliente envia uma mensagem criptografada para o servidor, que a descriptografa e envia de volta a mesma mensagem criptografada para o cliente.

A criptografia usada é o AES (Advanced Encryption Standard) no modo CBC (Cipher Block Chaining). O AES é um algoritmo de criptografia simétrica amplamente utilizado, que permite tanto a criptografia quanto a descriptografia usando a mesma chave.

* O cliente define a chave de criptografia KEY e o endereço do servidor (HOST e PORT).
* A função encrypt(plaintext) é responsável por criptografar a mensagem. Ela utiliza a chave KEY e o modo de operação CBC. A mensagem é preenchida com padding (preenchimento) para que tenha um tamanho múltiplo do tamanho do bloco do AES. O vetor de inicialização (IV) é gerado aleatoriamente pelo AES e é concatenado com o ciphertext (texto criptografado).
* O cliente cria um socket, se conecta ao servidor e envia o ciphertext.
* O servidor espera por uma conexão, aceita a conexão do cliente e inicia um loop para receber os dados.
* O servidor recebe o ciphertext do cliente.
* A função decrypt(ciphertext) é responsável por descriptografar o ciphertext recebido.
* Ela extrai o IV dos primeiros 16 bytes do ciphertext e o ciphertext restante para descriptografar usando a chave KEY e o modo CBC. O padding é removido do plaintext descriptografado.
* O servidor exibe a mensagem descriptografada no console e envia de volta o ciphertext recebido para o cliente.
* O cliente recebe o ciphertext do servidor.
* O cliente chama a função decrypt(ciphertext) para descriptografar o ciphertext recebido. O IV é extraído dos primeiros 16 bytes do ciphertext e é usado junto com a chave KEY e o modo CBC para descriptografar o restante do ciphertext. O padding é removido e o plaintext é exibido no console.

Dessa forma, o cliente e o servidor são capazes de se comunicar de forma segura através da criptografia AES no modo CBC. É importante lembrar que a segurança da criptografia depende da proteção adequada da chave de criptografia.

**4. Implementar um programa de chat simples usando sockets TCP. Os alunos podem criar um servidor que aceita várias conexões de clientes e encaminha as mensagens de um cliente para todos os outros clientes conectados.**

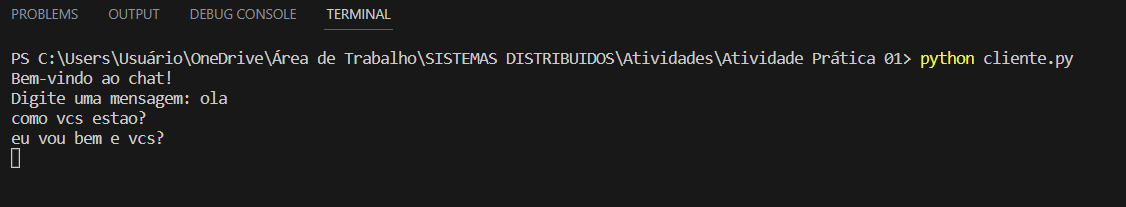
**Os arquivos do servidor e do cliente são: servidor.py; cliente.py**

O servidor cria um socket e escuta em um endereço IP e porta especificados. Quando um cliente se conecta, uma nova thread é iniciada para lidar com a conexão desse cliente. Cada thread aguarda por mensagens do cliente conectado e encaminha a mensagem recebida para todos os outros clientes conectados.

O cliente cria um socket e se conecta ao endereço IP e porta do servidor. Uma thread separada é iniciada para receber mensagens do servidor. O cliente aguarda a entrada do usuário e envia as mensagens para o servidor.

Para executar o programa vamos abrir 1 terminal no próprio vscode para inicializar o servidor através: python servidor.py

E outros terminais para os clientes e inicializar os clientes através: python cliente.py

exemplo do código funcionando: