Atividade 2: Cliente e Servidor TCP

***** Data de entrega: 16 de setembro de 2025*

**** Este trabalho deve ser realizado utilizando a linguagem C. Trabalhos que utilizarem bibliotecas externas poderão receber nota zero na implementação, pois todos os trabalhos serão compilados utilizando o padrão destas linguagens. Isto vale para quem utiliza Windows, certifique-se de compilar e executar em máquinas com Linux/Unix antes de entregar a tarefa.*

**** Lembre-se de justificar e comprovar suas respostas no relatório.*

**** **Detalhes da implementação:***

O código não pode apresentar nenhum warning quando compilado. Para cada warning exibido na compilação será descontado 10% da nota deste trabalho. (Certifique-se de compilar seu código com a flag -Wall do gcc).

Adicione todos os comentários necessários no código para indicar o que foi feito para realizar as questões e identificar as diferentes funções.

**** Entregáveis: relatório e códigos. Regra para atribuição de nota: Relatório 50% e Código 50%.*

Arquivos iniciais fornecidos

- servidor.c: servidor TCP que:
  - faz bind em 127.0.0.1
  - entra em listen,
  - para cada conexão aceita, envia um banner “Hello from server! + time” e fecha a conexão do cliente, mas permanece escutando para o próximo cliente
  - grava um arquivo server.info com IP/PORT
- cliente.c: cliente TCP que:
  - aceita IP/PORT via argumentos ou lê de server.info,
  - conecta, lê apenas o banner do servidor e fecha a conexão.

Esses dois códigos são apenas o **esqueleto inicial** para rodar e observar; todas as questões abaixo pedem que você evolua os programas partindo dessa base.

1. Analise os códigos dos programas cliente.c e servidor.c e explique as funções usadas para comunicação via socket (socket, bind, listen, accept, connect, read, write). Procure nas páginas de manual do Linux, a descrição das funções relacionadas ao uso de *sockets*.

2. Explique o Three-Way Handshake do TCP e como socket, bind, listen (servidor) e connect (cliente) participam. Quais mensagens são trocadas e em que ordem?

3. Compile e execute os programas cliente.c e servidor.c em uma mesma máquina.
   gcc -Wall cliente.c -o cliente
   gcc -Wall servidor.c -o servidor
   Houve algum erro com a função bind? Em caso afirmativo, qual a sua causa?

Se necessário, modifique os programas de forma que este erro seja corrigido e informe quais modificações foram realizadas.

4. Altere o código do servidor para ser automatizada a escolha da porta, mantendo 127.0.0.1 como IP. Após o bind, use getsockname() para descobrir a porta real e:
    - imprimir "[SERVIDOR] Escutando em 127.0.0.1:<PORTA>",
    - atualizar server.info com IP=127.0.0.1 e PORT=<PORTA>.
5. Defina uma função que mostre a 4-tupla de dados que caracteriza a conexão.

    No cliente: após connect, chame getsockname() e imprima:
    “local: <IP_local>:<porta_local>”.
    No servidor: após accept, chame getpeername() e imprima:
    “remoto: <IP_remoto>:<porta_remota>”.

    Mostre capturas de tela/saídas comprovando que os endpoints batem com as linhas do seu programa.

6. Faça envio de mensagens entre cliente e servidor:

    No cliente: leia uma linha do stdin e envie ao servidor.
    No servidor: receba e imprima a mensagem na saída padrão (como “[CLI MSG] …”).
    Pode ler uma vez com read() e imprimir; para mensagens curtas funciona bem.

7. Com ferramentas do sistema como ss ou netstat, execute o cliente e servidor e, antes, durante, e depois da conexão, colete evidências sobre os sockets. Inclua no relatório o trecho relevante e explique como a 4-tupla na ferramenta corresponde às impressões do seu programa.

8. Faça uma captura curta do handshake e da troca de dados no loopback:
    sudo tcpdump -i lo 'tcp[tcpflags] & (tcp-syn) != 0'

    Salve o trace e inclua screenshots do Wireshark com breve explicação e as observações onde é evidenciado o 3way-handshake.

9. Conecte com telnet ou nc ao seu servidor e descreva o que acontece.
    Dá para usar essas ferramentas no lugar do seu cliente.c? Justifique e comprove com evidências (e.g., echo de uma linha).

Entrega no classroom com três arquivos (não coloque .zip):

- Códigos comentados: cliente.c, servidor.c
- Relatório contendo:
    - respostas das partes teóricas,
    - descrição clara das modificações feitas em cada questão (arquivo/linhas),
    - evidências (ss/netstat, prints do terminal, screenshots de wireshark, etc),