Guião 06

1.2

No código-fonte do servidor, ele está sempre a aguardar uma mensagem de um usuário no endereço IPv4 0.0.0.0, na porta UDP 5005.

No código-fonte do cliente, há duas funções para interação com o usuário, e o seu endereço IPv4 é 127.0.0.1, usando a mesma porta UDP que o servidor. Em seguida, ele aguarda uma mensagem do usuário para enviá-la ao servidor usando esse endereço IP e porta, para que o servidor receba a mensagem.

No Wireshark, para cada mensagem, podemos capturar um pacote UDP que contém as informações da mensagem (tamanho, conteúdo) e também informações sobre as portas.

2.2

Após analisar o código-fonte do servidor, definimos o IP e a porta do servidor (iguais aos anteriores). Depois disso, o socket é criado e começa a ouvir clientes que tentam se conectar (máximo de 5).  
Quando um cliente se conecta, o servidor armazena o endereço e o socket do cliente conectado.  
Agora, ele cria uma *thread* para ouvir mensagens, enquanto ainda pode aceitar mais clientes.  
Ao receber uma mensagem de um cliente, ele devolve a mensagem (faz o *echo*) para o cliente.

No cliente, ele define seu IP e porta (iguais aos anteriores) e cria um socket, que se conecta ao servidor. Em seguida, ele aguarda uma mensagem para enviá-la ao servidor usando o endereço IP e porta, para que o servidor receba a mensagem (que está constantemente ouvindo). Por fim, antes de permitir que o usuário envie outra mensagem, aguarda que o servidor faça o *echo* da mensagem enviada.

No Wireshark, capturamos 3 pacotes TCP quando o cliente se conecta ao servidor:

1. Primeiro, há uma sincronização (SYN) do cliente para o servidor solicitando conexão.
2. Depois, o servidor envia um pacote de sincronização/acknowledge (SYN/ACK) para confirmar a conexão e reconhecer as informações do cliente.
3. Por fim, o cliente envia um pacote de confirmação (ACK) de volta ao servidor para confirmar que recebeu as informações.

Capturamos mais 4 pacotes TCP por troca de mensagem na seguinte ordem:

1. Um pacote push/acknowledge do cliente para o servidor, que inclui as informações da mensagem e solicita que o servidor confirme o recebimento.
2. Um pacote acknowledge do servidor para o cliente informando que a mensagem foi recebida corretamente.
3. Um pacote push/acknowledge do servidor para o cliente devolvendo a mensagem enviada e solicitando que o cliente confirme o recebimento.
4. Um pacote acknowledge do cliente para o servidor confirmando que recebeu a mensagem de volta.