UNIVERSIDADE FEDERAL DO RIO GRANDE DO NORTE
CENTRO DE TECNOLOGIA
DEPARTAMENTO DE ENGENHARIA DE COMPUTAÇÃO E AUTOMAÇÃO

Roteiro de Aula Prática – Estudo dos protocolos TCP e UDP	
DISCIPLINA:	DCA0130 – Redes de Computadores
PROFESSOR:	Carlos Manuel Dias Viegas

Esta prática consiste em uma introdução aos protocolos TCP e UDP por meio de programação com Sockets.

- Os requisitos para a realização desta prática são a instalação do Python na versão 3 e ter assistido às videoaulas sobre os protocolos TCP e UDP disponibilizadas no SIGAA;
- Esta prática consiste em realizar as tarefas descritas abaixo e responder às questões propostas neste documento;
- Este documento, com as devidas respostas, deverá ser submetido em uma tarefa específica no SIGAA até o dia **14/07/2021**;
- Esta prática deve ser realizada em duplas, podendo ser formadas por alunos de diferentes turmas da disciplina de redes de computadores (DCA0130) do semestre 2021.1.

Os códigos fonte (em Python) necessários para iniciar esta prática estão disponíveis na seguinte página:

`https://www.dca.ufrn.br/~viegas/disciplinas/DCA0130/files/Sockets/`

Nome do discente (1): Francisco Daniel Davi Turma: T02

Nome do discente (2): Luiz Henrique Araújo Dantas Turma: T01

Tarefa A: Experimento TCP

Faça o download do `clienteTCP.py` e `servidorTCP.py` na página indicada (acima).

1. Execute o Wireshark, inicie a captura e aplique o filtro para TCP na porta 65000: `tcp.port == 65000`
2. Execute o `servidorTCP` e conecte o `clienteTCP` ao mesmo (lembre de editar o IP e as portas conforme necessário para esta tarefa);
3. Envie uma mensagem ao servidor e aguarde a resposta;
4. Pare a captura de pacotes no Wireshark;
5. Analise os pacotes capturados e responda aos itens abaixo:

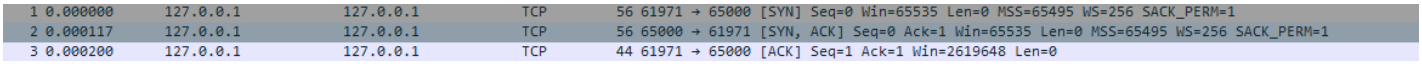
Responda às questões abaixo:

a) Quantas mensagens foram “trocadas” entre o cliente e o servidor antes da sua mensagem ser propriamente enviada?

3 mensagens.

b) Que mensagens são estas? (falar das *flags/codebits*)

Dentre essas 3 mensagens as duas primeiras flags são do tipo SYN, que é utilizada para iniciar uma conexão entre o servidor e o cliente, a primeira mensagem possui a flag SYN para iniciar a conexão, a segunda mensagem possui as flags SYN + ACK, no qual envia uma requisição para se conectar e a confirmação, a terceira mensagem é finalmente a resposta do ACK pelo cliente.



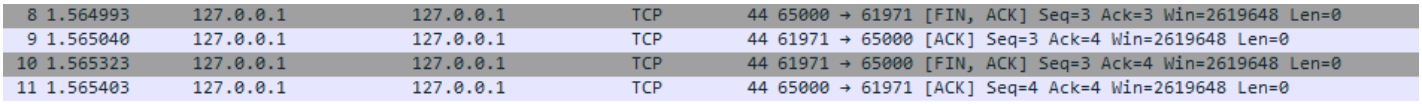
1	0.000000	127.0.0.1	127.0.0.1	TCP	56 61971 → 65000 [SYN] Seq=0 Win=65535 Len=0 MSS=65495 WS=256 SACK_PERM=1
2	0.000117	127.0.0.1	127.0.0.1	TCP	56 65000 → 61971 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=65495 WS=256 SACK_PERM=1
3	0.000200	127.0.0.1	127.0.0.1	TCP	44 61971 → 65000 [ACK] Seq=1 Ack=1 Win=2619648 Len=0

c) Qual o nome desse processo de troca de mensagens? E qual a sua finalidade?

O processo denomina-se “Three-way handshake”, ele é utilizado para iniciar uma conexão TCP, dado que o protocolo TCP é baseado em um envio seguro de mensagens a principal funcionalidade do Three-way handshake é realizar uma comunicação sólida para a troca de mensagens entre o cliente e o servidor.

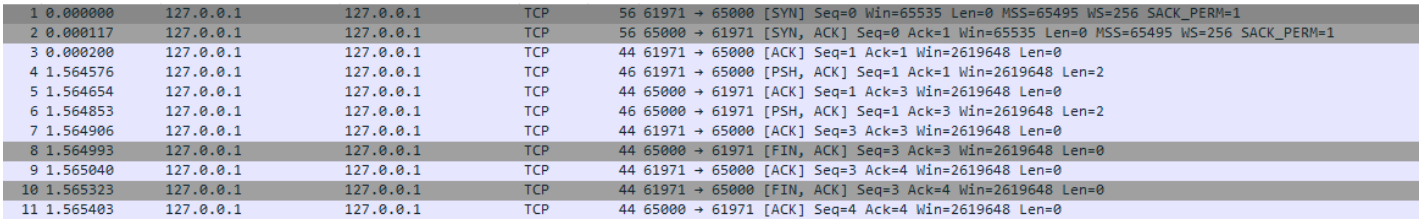
d) Como ocorreu o encerramento da conexão?

O encerramento da conexão foi feito a partir do “Four-way handshake”, no qual o servidor e o cliente iniciam uma troca de quatro mensagens com as flags FIN e ACK, eles encerram a comunicação independentemente, a sequência das mensagens são feitas primeiramente com o envio das flags FIN e ACK, e em seguida é enviado um ACK para que a conexão seja finalizada.



8	1.564993	127.0.0.1	127.0.0.1	TCP	44 65000 → 61971 [FIN, ACK] Seq=3 Ack=3 Win=2619648 Len=0
9	1.565040	127.0.0.1	127.0.0.1	TCP	44 61971 → 65000 [ACK] Seq=3 Ack=4 Win=2619648 Len=0
10	1.565323	127.0.0.1	127.0.0.1	TCP	44 61971 → 65000 [FIN, ACK] Seq=3 Ack=4 Win=2619648 Len=0
11	1.565403	127.0.0.1	127.0.0.1	TCP	44 65000 → 61971 [ACK] Seq=4 Ack=4 Win=2619648 Len=0

Relatório total de conexão entre o servidor e o cliente:



1	0.000000	127.0.0.1	127.0.0.1	TCP	56 61971 → 65000 [SYN] Seq=0 Win=65535 Len=0 MSS=65495 WS=256 SACK_PERM=1
2	0.000117	127.0.0.1	127.0.0.1	TCP	56 65000 → 61971 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=65495 WS=256 SACK_PERM=1
3	0.000200	127.0.0.1	127.0.0.1	TCP	44 61971 → 65000 [ACK] Seq=1 Ack=1 Win=2619648 Len=0
4	1.564576	127.0.0.1	127.0.0.1	TCP	46 61971 → 65000 [PSH, ACK] Seq=1 Ack=1 Win=2619648 Len=2
5	1.564654	127.0.0.1	127.0.0.1	TCP	44 65000 → 61971 [ACK] Seq=1 Ack=3 Win=2619648 Len=0
6	1.564853	127.0.0.1	127.0.0.1	TCP	46 65000 → 61971 [PSH, ACK] Seq=1 Ack=3 Win=2619648 Len=2
7	1.564906	127.0.0.1	127.0.0.1	TCP	44 61971 → 65000 [ACK] Seq=3 Ack=3 Win=2619648 Len=0
8	1.564993	127.0.0.1	127.0.0.1	TCP	44 65000 → 61971 [FIN, ACK] Seq=3 Ack=3 Win=2619648 Len=0
9	1.565040	127.0.0.1	127.0.0.1	TCP	44 61971 → 65000 [ACK] Seq=3 Ack=4 Win=2619648 Len=0
10	1.565323	127.0.0.1	127.0.0.1	TCP	44 61971 → 65000 [FIN, ACK] Seq=3 Ack=4 Win=2619648 Len=0
11	1.565403	127.0.0.1	127.0.0.1	TCP	44 65000 → 61971 [ACK] Seq=4 Ack=4 Win=2619648 Len=0

Tarefa B: Experimento UDP

Faça o download do `clienteUDP.py` e `servidorUDP.py` na página indicada (acima).

1. Execute o Wireshark, inicie a captura e aplique filtros para UDP na porta 65000: `udp.port == 65000`
2. Execute o `servidorUDP` e conecte o `clienteUDP` ao mesmo (lembre de editar o IP e as portas conforme necessário para esta tarefa);
3. Envie uma mensagem ao servidor e aguarde a resposta;
4. Pare a captura de pacotes no Wireshark;
5. Analise os pacotes capturados e responda aos itens abaixo:

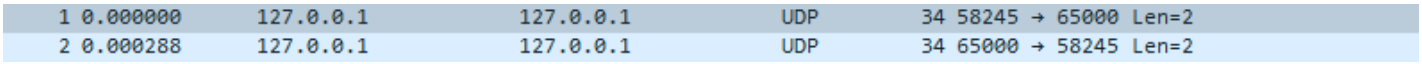
Responda às questões abaixo:

a) Alguma outra mensagem foi “trocada” entre o cliente e o servidor antes da sua mensagem ser enviada?

Não.

b) Quantas mensagens foram efetivamente capturadas?

Duas mensagens.



1	0.000000	127.0.0.1	127.0.0.1	UDP	34 58245 → 65000 Len=2
2	0.000288	127.0.0.1	127.0.0.1	UDP	34 65000 → 58245 Len=2

c) Houve troca de mensagens para encerramento da conexão?

Não.

d) Faça uma breve comparação entre o funcionamento do TCP e do UDP.

Comparando o funcionamento de TCP com o UDP podemos notar que a segurança de envio e recebimento de dados, ordenamento dos pacotes recebidos, são uns dos fatores mais importantes para o TCP, porém esses procedimentos têm o custo de um tempo maior para o fluxo de mensagens, enquanto no UDP não há nenhuma segurança de recebimento (mensagens podem nunca chegar) e ordenamento das mensagens serão recebidas ou enviadas na transmissão, no entanto a falta desses atributos favorecem o UDP na velocidade com que as informações são trocadas.