PTC-3450 - Exercício Programa 2 - 2017

GABARITO

Nesse exercício, você vai obter estimativas do *timeout* utilizado pelo TCP em uma troca de pacotes.
Documento da forma mais detalhada possível suas respostas. Para obter os gráficos, sugere-se o uso do Matlab, mas você pode usar qualquer outra ferramenta computacional.
Siga os seguintes passos:

a) (1.0 pontos)Escolha um servidor www, por exemplo `www.lcs.poli.usp.br`, `www.google.com.br`, ou qualquer outro que você quiser. Use o `ping` para obter 250 estimativas de RTTs, ou seja 250 valores para `SampleRTT`. De preferência, use uma rede Wi-Fi para obter os seus dados. Assim você terá maior variabilidade e seus resultados serão mais interessantes. Ignore os pacotes perdidos.

Como exemplo, considerei o acesso ao servidor web `www.mit.edu`. Os dados obtidos, em milissegundos, estão no arquivo `exemploep2.mat`.

b) (1.0 pontos) A partir dos valores `SampleRTT` obtidos, calcule 250 valores para `EstimatedRTT`, `DevRTT` e `TimeoutInterval`. Deixe claramente indicado a forma como você fez os cálculos e as condições iniciais adotadas.

Segue programa Matlab para os cálculos pedidos. Utilizou-se como condição inicial para `EstimatedRTT` o valor de `SampleRTT(1)`. Para `DevRTT` utilizou-se 0 como condição inicial. Note que não é necessário usar nenhum *loop*.

```
clear all; close all;
load emploep2.mat;
SampleRTT = times;
Npontos = length(SampleRTT);
alfa = 0.125;
beta = 0.25;
xic = filtic(alfa, [1 -(1-alfa)], SampleRTT(1));  %Condição inicial
EstimatedRTT = filter(alfa, [1 -(1-alfa)], SampleRTT, xic);
DevRTT = filter(beta, [1 -(1-beta)], abs(SampleRTT-EstimatedRTT));
TimeoutInterval = EstimatedRTT+ 4*DevRTT;
```

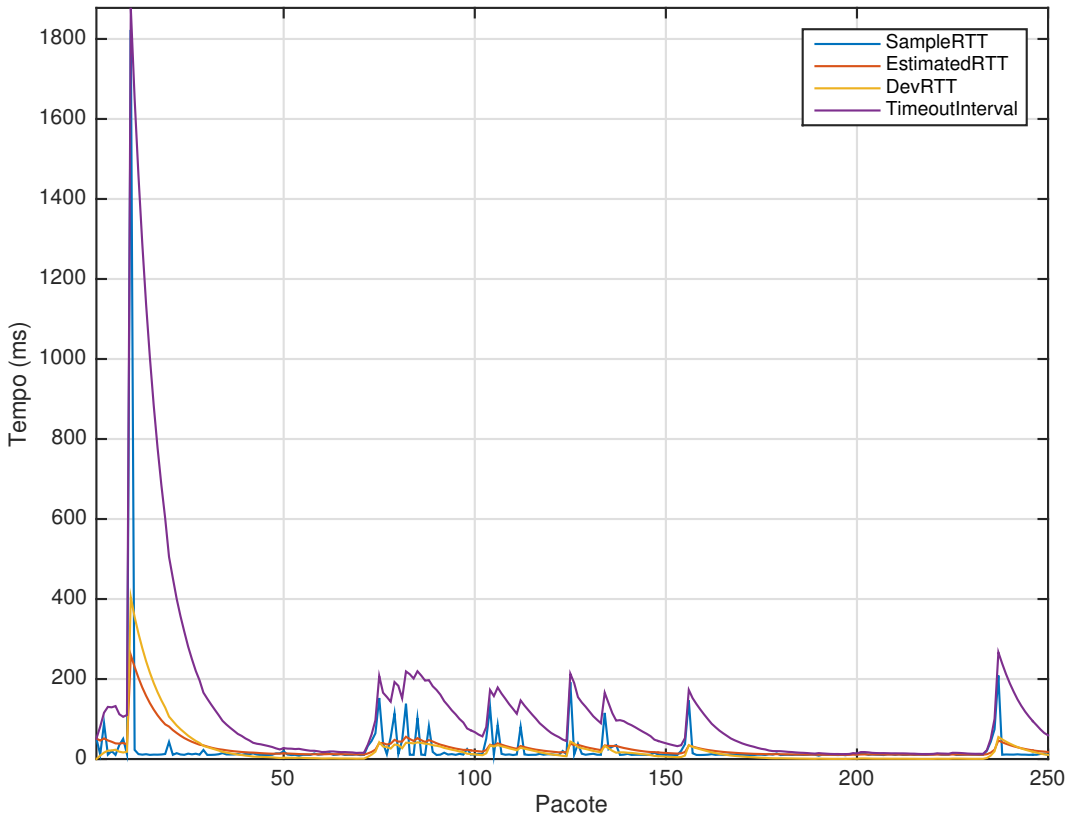
c) (1.0 pontos) Qual o valor médio dessas variáveis no período observado?

```
>> mean(SampleRTT)
ans =
    27.2093
>> mean(EstimatedRTT)
```

```
ans =
    28.1495
>> mean(DevRTT)
ans =
    24.3513
>> mean(TimeoutInterval)
ans =
    125.5548
```

d) (1.0 pontos) Apresente num mesmo gráfico curvas para `SampleRTT`, `EstimatedRTT`, `DevRTT` e
`TimeoutInterval`. Detalhe como você obteve o gráfico.

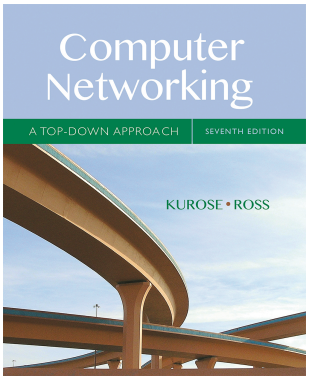
```
tempos = 1:Npontos;
plot(tempos,SampleRTT, tempos,EstimatedRTT, tempos,DevRTT, tempos,
TimeoutInterval);
legend('SampleRTT', 'EstimatedRTT', 'DevRTT','TimeoutInterval');
xlabel('Pacote');
ylabel('Tempo(ms)');
axis([1 Npontos 0 max(TimeoutInterval)]);
grid;
```



e) (1.0 pontos) Conclua comentando todos os resultados obtidos.

Note que num fluxo TCP real, quando o valor de `SampleRTT` ultrapassa o `EstimatedRTT` anterior, temos um *timeout*, ocasionando perda do pacote.

Wireshark Lab: UDP
SOLUTION



Supplement to *Computer Networking: A Top-Down Approach, 7th ed.,* J.F. Kurose and K.W. Ross

---

The answers below are based on the trace file http-ethereal-trace-5, in http://gaia.cs.umass.edu/wireshark-labs/wireshark-traces.zip

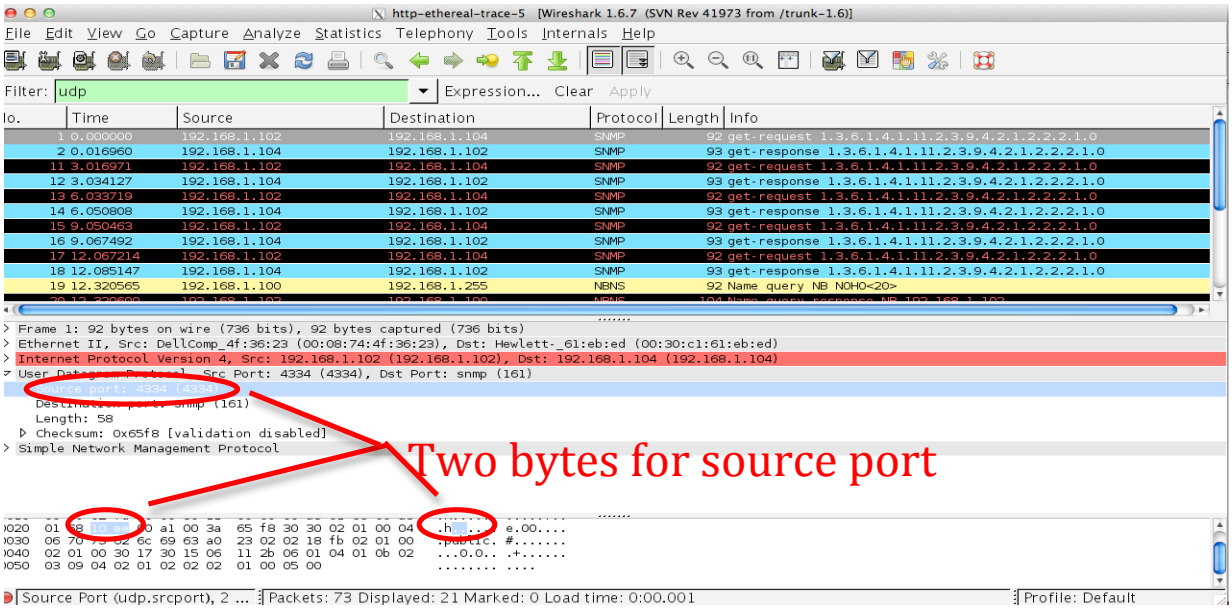
Here is what is printed out for packet 1 in this trace:

```
No.     Time        Source                Destination
Protocol Length Info
      1 0.000000    192.168.1.102         192.168.1.104         SNMP
92     get-request 1.3.6.1.4.1.11.2.3.9.4.2.1.2.2.2.1.0

Frame 1: 92 bytes on wire (736 bits), 92 bytes captured (736 bits)
Ethernet II, Src: DellComp_4f:36:23 (00:08:74:4f:36:23), Dst:
Hewlett-_61:eb:ed (00:30:c1:61:eb:ed)
Internet Protocol Version 4, Src: 192.168.1.102 (192.168.1.102),
Dst: 192.168.1.104 (192.168.1.104)
User Datagram Protocol, Src Port: 4334 (4334), Dst Port: snmp (161)
    Source port: 4334 (4334)
    Destination port: snmp (161)
    Length: 58
    Checksum: 0x65f8 [validation disabled]
Simple Network Management Protocol
```

1. Select *one* UDP packet from your trace. From this packet, determine how many fields there are in the UDP header. (You shouldn't look in the textbook! Answer these questions directly from what you observe in the packet trace.) Name these fields. *Answer: There are four fields in the header: source port, destination port, Length, and checksum.*

2. By consulting the displayed information in Wireshark's packet content field for this packet, determine the length (in bytes) of each of the UDP header fields. *Answer: by clicking on the source port field (top red circle in the figure below), we see the value corresponding to that port number value in the packet content window at the bottom of the Wireshark display. The port number is shown as a hexadecimal number (small lower left red circle) and in ASCII format (small lower right red circle), and is two bytes long.*



3. The value in the Length field is the length of what? (You can consult the text for this answer). Verify your claim with your captured UDP packet. *Answer: The UDP length field is the length of the header and data fields of the UDP segment, measured in bytes).” The displayed packet has a length field of 58 bytes. We know there are 8 byres of header. If we look at the packet content field, we also find 50 bytes of hexadecimal or ASCII-encoded data which corresponds to the payload of this UDP segment.*
4. What is the maximum number of bytes that can be included in a UDP payload? (Hint: the answer to this question can be determined by your answer to 2. above). *Answer: Since there are only 16 bits, the maximum length of a UDP segment (including header) is 2**16 -1 or 65535 bytes.*
5. What is the largest possible source port number? (Hint: see the hint in 4.) *Answer: Since there are only 16 bits, the maximum source port number is 2**16 -1 or 65535 bytes.*
6. What is the protocol number for UDP? Give your answer in both hexadecimal and decimal notation. To answer this question, you’ll need to look into the Protocol field of the IP datagram containing this UDP segment (see Figure 4.13 in the text, and the discussion of IP header fields). *Answer: UDP has a protocol number of 17 (this number is displayed in Wireshark as the value of the “protocol:” field in the IPV4 datagram.*
7. Examine a pair of UDP packets in which the first packet is sent by your host and the second packet is a reply to the first packet. Describe the relationship between the port numbers in the two packets. *Answer: Let’s look at packets 1 and 2 in the trace. These packets carry SNMP application-level messages encapsulated inside of UDP. (Got that? If not, re-read section 1.5 and Figure 1.24 in the text, in particular) The IP address of the sender of packet 1 is the IP address of the destination of packet 2, and the IP address of the destination of packet 1 is the IP*

address of the sender of packet 2, The names in the Info field of get-request and get-response suggest that the second packet (a response) is sent in reply to the first packet (a request). Indeed this is the case. The value of the source port in packet 1 is the value of the destination port of packet 2; the value of the destination port of packet 1 is the value of the source port of packet 2,