

Lista 03 - Camada de transporte.

Perguntas de Revisão -

(R4) - Algumas aplicações necessitam de alto nível de confiabilidade e não necessitam da entrega confiável do TCP, o que faz com que o UDP seja uma ótima escolha. Além disso, o TCP pode limitar a taxa de envio em momentos de congestionamento, o que pode ser indesejado, e depende das aplicações.

(R6) - Sim, a transferência confiável de dados pode ser constituida ou adicionada diretamente na camada de aplicação, porém isso não é tão simples (mas possível).

(R7) - Sim, todos os 2 segmentos são para o mesmo destinatário, o SO (sistema operacional) forma o endereço IP para os terminais de onde vieram cada um dos segmentos.

R15 -

- a) O primeiro segmento possui 20 bytes
- b) O número de reenviamento será 90.

P3 -

Soma dos dois primeiros é:

$$\begin{array}{r} 01010011 \\ 01100110 \\ \hline 10111001 \end{array} \quad \rightarrow \quad \begin{array}{l} \text{Soma da soma com a terceira.} \\ 10111001 \\ 10011100 \\ 01110110 \\ \hline 10101100 \\ \Delta + 1 \\ \boxed{00101110} \end{array}$$

Realizando o complemento de 1:

$$11010001$$

Para detecção de erros, desconsiderando os bits de paridade (as 3 enviadas, e as de verificação), se houver um erro somente, haverá algum erro.

Todos os erros de um (1) bit são detectados, porém erros de 2 bits podem passar despercebidos (circunferente somam 1).

P25

- a) O TCP, por exemplo, pode colocar mais ou menos do que somente uma mensagem no seu segmento. Já o UDP, coloca em um segmento o que quer que seja enviado, ou seja, se aplicar em uma UDP uma mensagem, o segmento UDP contém aquela mensagem, logo, com o UDP, a aplicação controla quais dados são enviados em cada segmento.

b) Devido ao TCP possuir em sua notificação controle de fluxo e controle de congestionamento, que pode gerar atrasos, o que não está presente no UDP.

P26

- a) Sabendo que o campo de número de sequência TCP possui 4 bytes = 32 bits, isso gera $2^{32} = 4.294.967.296$ números de sequências diferentes. Além disso, o número de sequência incrementa de acordo com o número de bytes enviados, logo é fixo e de $2^{32-4=28} = 268.435.456$.

b) Para calcular o número de segmentos:

$$\frac{2^{32}}{53.6} = 8.012.999 \text{ segmentos}$$

Para cada segmento, são adicionados 66 bytes, assim $66 \cdot 8.012.999 = 528.185.7934$ bytes da cabeca →

O número total de bytes transmitidos é de:

$$\underbrace{2^{32}}_{\text{dados}} + \underbrace{528,857,931}_{\text{Cabeçalho}} \approx 4,824 \cdot 10^9 \text{ bytes}$$

total

Por descobrir o tempo gasto, tem-se:

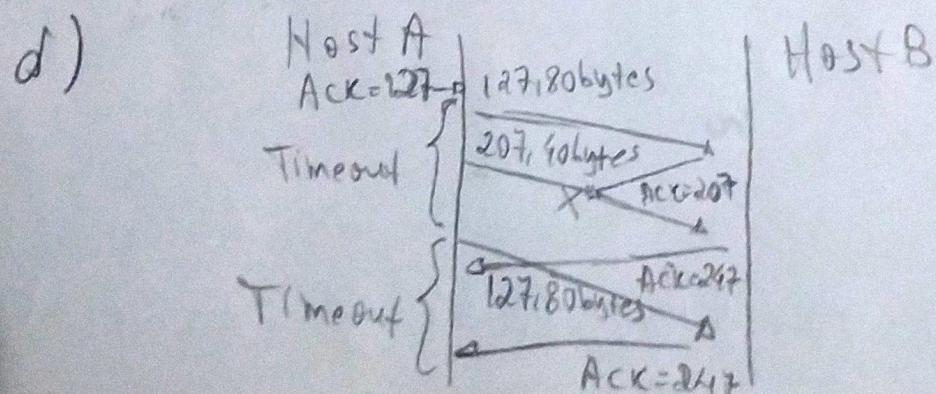
$$\frac{4,824 \cdot 10^9}{155 \cdot 10^{-6}} = 250 \text{ segundos}$$

(P27) -

a) No segundo segmento, o número de sequência é 207 (é o primeiro byte sequencial + 40 bytes) e o número da porta de origem é 302 e destino é 80.

b) Número de ACK é 207, origem é porta 80 e destino é porta 302.

c) Se o segundo chegar primeiro, o ACK é 127, pois ainda está aguardando o 127 byte.



(P33) - $\text{RTT}_{\text{Estimado}} = (1-\alpha) \times \text{RTT}_{\text{Estimado}} + \alpha \times \text{RTT}_{\text{medido}}$

 $\text{RTT Desvio} = (1-\beta) \times \text{RTT Desvio} + \beta \times |\text{RTT}_{\text{medido}} - \text{RTT}_{\text{estimado}}|$
 $\text{Timeout} = \text{RTT}_{\text{estimado}} + 4 \times \text{RTT Desvio}$
 $\alpha = 0,125 \quad | \quad \beta = 0,25 \quad | \quad \downarrow^{\circ} \text{ RTT}_{\text{estimado}} = 300\text{ms}, \downarrow^{\circ} \text{ DevRTT} = 5\text{ms}$

$2^{\circ} \text{ } \text{RTT}_{\text{Estimado}} = (1-0,125) \times 300\text{ms} + (0,125) \times 306\text{ms} = 306,75\text{ms}$

$2^{\circ} \text{ DevRTT} = (1-0,25) \times 5 + 0,25 \times |306 - 300,75| = 5,0625\text{ms}$

$2^{\circ} \text{ Timeout} = 300,75 + 4 \times 5,0625 = 325\text{ms}$

; ↓

Assim:

Samp k	Estimated RTT	DevRTT	Timeout
106	300,75	5,0625	325 ms
120	303,15625	8,0078	335,1875 ms
140	307,7617	14,0654	349,0234 ms
90	305,15415	14,4345	343,2793 ms
115	306,7238	12,8949	358,3053 ms

- (P40) a) Slowstart de 5 até 6 e de 23 a 26.
- b) Nos intervalos de 6 ate 16 e de 17 ate 22
- c) Após o 16º rebole, quando é detectado por 3 ACKs duplicados, esse faz o timeout permanecer para 1.

d) Após a 2^a rodada, a porta é detectada por timeout, e a janela ajustada para 1.

e) O limite inicial é de 32.

f) O limite é 25. (metodo de quando se consegue)

g) O limite é 14 (metodo de quando se consegue).

h) A cada rodada soma 6, dobrando o limite.

$$1^{\circ} \rightarrow 1 \rightarrow 2^{\circ} = 2 \rightarrow 3^{\circ} = 4 \rightarrow 4^{\circ} = 8 + 5^{\circ} = 16$$

6^o = 32, e no 7^o permanece 32 (congestionamento).

Assim, 1+2+4+8+16+32 = 63 até a 6^a rodada

- mas 7^o = 64+32 = 96, logo o 7^o pacote é enviado
depois da 7^o rodada.

i) O valor do ssthresh sera o metodo do valor atual
 $\frac{8}{2} = 4$, e o tamanho da janela é o metodo do ssthresh
mas os 3 pacotes já entregues = 7 = (4+3)

K) Com todos os tempos:

$$17 - 1 \rightarrow 18 - 2 \rightarrow 19 - 4 \rightarrow 20 - 8 \rightarrow 21 - 16$$

e $22 - 25$, tudo lizado!

$$1 + 2 + 4 + 8 + 16 + 21 = 52$$
 pacotes.

(P55)

- a) A resposta é enviada para o servidor Y.
- b) Sim, o servidor pode ter certeza que o cliente está em Y, pois em caso de falsificação, o SYNACK seria enviado para Y e o TCP ACK não seria retornado pelo host. Mesmo se enviado um ACK corretamente, não saberia o número de sequência correto do servidor.