MAC 448 – PRC
BCC – 1º Semestre de 2009
Exercícios – Camada de Transporte – Parte 2

1. Explique resumidamente o significado das seguintes características gerais do TCP: ponto-a-ponto; transporte confiável; pipeline; full duplex; orientado a conexão;fluxo controlado; congestionamento controlado

Ponto-a-ponto – O TCP é unicast e não multicast. O ponto A se comunica com o ponto B e vice versa.

Transporte confiável – o TCP tem controle de erro e retransmissões em caso de erro. Se houver erro ele é recuperável pelo protocolo.

Pipeline – são enviados vários segmentos antes de receber uma ACK.

Full duplex – os dois lados enviam e recebem ao mesmo tempo.

Orientado a conexão – antes de começar o fluxo de dados, há um pedido de conexão. Após o aceito da conexão é feita a transferência de dados.

Fluxo controlado – o receptor pode pedir para que o transmissor pare de enviar dados.

Congestionamento controlado – o transmissor detecta que há excesso de tráfego na rede e diminui a quantidade de dados enviados.

2. No header do pacote TCP, qual o significado do número de sequência do pacote (32 bits) e o número de ACK (32 bits)?

Número de sequência do pacote (32 bits) – contém o número do primeiro byte do pacote sendo enviado. Assim, o lado que recebe sabe que o próximo pacote terá que ter neste campo, o valor anterior mais o comprimento do pacote recebido.

Número de ACK (32 bits) – contém o número do próximo byte que o lado que enviou, espera receber. Isto é, todos os anteriores já foram recebidos bem.

3. No header do pacote TCP, qual o significado do campo RcvWindow?

Este campo informa ao receptor da mensagem qual o tamanho da área

disponível para receber dados do transmissor. Assim, dependendo do tamanho da próxima mensagem a ser enviada, o receptor deste segmento pode avaliar se pode enviar ou não o próximo pacote.

4. Explique a técnica de piggyback (carona) no TCP.

Supondo que X envia uma mensagem para Y.
Y deve confirmar que recebeu o pacote. Esta confirmação pode pegar carona em uma mensagem que Y tenha que enviar para X.

5. O que é controle de fluxo e como isso é feito no TCP?

Controle de fluxo é uma iniciativa do lado receptor. O lado receptor, por não ter capacidade de receber mais dados (não existem buffers suficientes, por exemplo) envia uma mensagem ao transmissor, solicitando que pare de enviar dados. O envio é retomado tão logo o receptor avise novamente a origem que pode continuar. Na verdade, o receptor informa qual a memória disponível para recber mais dados. Se for pequena, o transmissor conclui que não deve enviar mais nada enquanto não for informado que existe memória suficiente para continuar enviando.

6. O que é controle de congestionamento e como isso é feito no TCP?

Controle de congestionamento são ações que o TCP toma durante a operação, com o objetivo de evitar que a carga de tráfego que flui por um dado link, exceda a capacidade do mesmo.
O TCP começa com baixa carga, ou seja, tamanho pequeno de janela e vai aumentando até que o canal dê sinal de estar lotado (perda de dados ou timeout). A partir daí, diminue o tamanho da janela e vai aumentando de novo. A idéia é ocupar ao máximo a capacidade do canal, mas com cuidado de não excedê-la.

7. O controle de fluxo e o controle de congestionamento têm o mesmo objetivo?

Sim, o objetivo é o mesmo. Não exceder a capacidade da rede.

8. Se sim, qual é esse objetivo. Se não, qual o objetivo de cada um deles?

Evitar que sejam enviados mais dados que a rede pode suportar.
Portanto evitar a perda de dados.

9. A iniciativa para cada um destes controles é tomada pelo lado origem ou pelo lado destino de cada conexão?

Controle de fluxo – iniciativa do lado destino. O lado destino, solicita ao lado origem que não envie dados. Isso é feito, informando que não há buffer para receber os dados. O campo RcvWindow (janela de recepção) dos segmentos TCP é informado pelo lado que está recebendo. O lado origem só envia mais se: **LastByteSent – LastByteAcked <= RcsWindow**

Controle de congestionamento – iniciativa do lado origem. O lado origem verifica que há a ocorrência de time-outs ou triplo acks para os mesmos pacotes. Isso é um indício de que a rede sobrecarregada e perdendo informação. Então o lado origem toma a iniciativa de enviar menos dados, diminuindo o tamanho da janela.

10. Quantas mensagens são trocadas no TCP para estabelecer uma conexão e quais são elas?

São 3 mensagens:

- Lado cliente envia msg com o bit SYN do campo de flags ligado, especificando o número inicial de seqüência para esta conexão.
- Lado servidor responde com mensagem de ACK e especifica o número inicial de sequência para o cliente nesta conexão.
- Lado cliente responde com ACK. Esse último ACK já pode ser enviado com um pacote de informação.

11. Quantas mensagens são trocadas no TCP para fechar uma conexão e quais são elas?

São 4 mensagens:

- Lado cliente envia com o bit FIN do campo de flags ligado.
- Lado servidor responde com ACK.
- Lado servidor envia msg com o bit FIN ligado.
- Lado cliente responde com ACK.

12. Quais são as 2 maneiras de tratar o congestionamento? Qual a maneira usada pelo TCP?

- Observando o comportamento da rede
  - É o caso do TCP (perda de pacotes – triple ACK e timeout)

- Informação vinda da camada de rede – basta 1 só bit de informação – pode ser feito de 2 maneiras
  - o elemento congestionado informa expontaneamente
  - envia-se um pedido de informação
  - Exemplo – ATM ABR – Available Bit Rate

13. Quais as 2 fases em cada conexão usadas pelo TCP para controlar o congestionamento?

- 2 fases distintas – em cada conexão – a variável CongWin (Congestion Window) e a variável Limiar (Threshold) controlam o processo.

  - Partida lenta – CongWin = 1, 2, 4, ... até o Limiar

  - Prevenção de congestionamento - CongWin++ até haver alguma perda.
    - 3 ACKs repetidos: Limiar = CongWin/2; CongWin = Limiar;
    - Time-out:
      - Tahoe:Limiar=Congwin/2; Congwin=1; go to Partida lenta
      - Reno:Limiar=CongWin/2; CongWin=Limiar

14. Dê o algoritmo resumido para cada uma das fases.

```
Fase Partida Lenta:
        CongWin = 1;
        for (cada segmento ACKed)
              Congwin=CongWin*2
        until (ocorrer uma perda OR
                CongWin > Limiar);
        para a fase seguinte;

Fase Prevenção de Congestionamento:
        until (ocorra uma perda) {
          A cada w segments ACKed: Congwin++
        }
        // ocorreu perda
        Se (perda = 3 ACKs repetidos) {
```

```
      CongWin/2; CongWin = Limiar;
}
Senão
 Se (perda= time-out) {
    TCP Tahoe:{
      Limiar = Congwin/2;
      Congwin = 1;
      retorne a partida lenta;
    }
    TCP Reno: {
      Limiar = Congwin/2;
      Congwin = Limiar;
      Continue nesta fase;
    }
```

15. O objetivo dessas duas fases é só controlar o congestionamento? Se não é, diga outro(s) objetivos.

O objetivo é principalmente controlar o congestionamento.
Porém o que ocorre é que com isso o TCP vai aumentando a ocupação do canal (aumentando a janela) tanto quanto for possível. A idéia é usar todo o recurso disponível dentro de uma conexão para acelar mais a transferência de dados.

16. Porque é necessário uma boa estimativa do time-out e como o TCP faz essa estimativa?

Se o timeout for pequeno, corre-se o risco de receber um timeout antes que os pacotes possam ser confirmados. Se o timeout for grande corre-se o risco de esperar demais. Daí a necessidade de uma boa estimativa.
O TCP mantém um valor estimado dinamicamente, baseado no tempo de resposta corrente da rede RTT:

- **RTT = (1-α)*RTT + α*último RTT (Média EWMA)**
- **Timeout = RTT + 4*Desvio**
- **Desvio = (1- β)*Desvio + β *|último RTT-novo RTT|**

O valor usual de α é 0,125 e o de β é 0,25

17. Porque e como o cálculo dinâmico do RTT privilegia as amostras mais recentes?

Basta desenvolver a fórmula acima e ver que o peso das amostras mais antigas vai decrescendo em valor.
É razoável que isso aconteça, pois a idéia é refletir a melhor estimativa do tempo de resposta. As amostras mais recentes têm mais chance de ser representativas que as mais antigas.