Fundação CECIERJ - Vice Presidência de Educação Superior a Distância

Curso de Tecnologia em Sistemas de Computação
Disciplina: Redes de Computadores I
AP2 - 1° semestre de 2008
Gabarito

1. **(1,0 ponto)** Um processo que executa no hospedeiro **A** está associado à porta **p**, e um processo que executa no hospedeiro **B** esta associado à porta **q**. É possível que datagramas UDP oriundos desses dois processos sejam entregues no mesmo *socket,* que tem número de porta **r**, existente no hospedeiro **C**? Explique.

   Resposta: Sim, ambos os segmentos serão direcionados para o mesmo *socket* no hospedeiro C. Para cada segmento recebido, na interface do *socket*, o sistema operacional disponibilizará para o processo o endereço IP de forma a determinar as origens dos segmentos individuais.

2. **(1,5 pontos)** Qual razão da existência de "temporizadores" *(timers),* nos protocolos para transferência confiável de dados *(reliable data transfer – rdt)?*

   Resposta: Os temporizadores servem para tratar as perdas de pacotes pelo canal de transmissão. Se uma confirmação para um pacote enviado, não é recebido, dentro do intervalo usado na temporização, então o pacote (ou seu ACK ou NAK) é considerado perdido e o pacote é retransmitido. Obviamente esta estratégia cria a possibilidade de pacotes em duplicata no receptor, mas este problema é resolvido através dos números de seqüência.

3. **(1,5 pontos)** Qual razão da existência de "números de seqüência", nos protocolos para transferência confiável de dados *(reliable data transfer – rdt)?*

   Resposta: Os "números de seqüência" são utilizados para que o receptor rdt possa detectar se um pacote que acaba de ser entregue contém dados novos ou se trata-se de um pacote retransmitido. Se o protocolo faz transferência confiável de dados com paralelismo (*pipelined),* como é caso do protocolo Repetição Seletiva, os números de seqüência servem também para que o receptor possa detectar pacotes perdidos ou recebidos fora de ordem. Nesse caso o receptor rdt deve preencher as lacunas no fluxo de *bytes* recebidos, antes da entrega dos dados para a aplicação.

4. **(2,0 pontos)** Uma confirmação TCP perdida não necessariamente força uma retransmissão. Explique por quê.

   Resposta: Por causa do uso da confirmação cumulativa. Ou seja, o recebimento da confirmação ***n*** pelo transmissor TCP, indica que todos ***n - 1*** *bytes* anteriores do fluxo de *bytes*, foram recebidos no receptor TCP. Essa confirmação ***n*** pode estar confirmando *bytes* contidos em um ou mais segmentos, cujas confirmações foram perdidas (ou não enviadas).

5. **(0,5 pontos cada item)** Suponha que o hospedeiro A envia dois segmentos para o hospedeiro B em uma conexão TCP. O primeiro segmento tem número de seqüência 110 e o segundo tem número de seqüência 140.

   a. Quantos *bytes* de dados estão contidos no primeiro segmento? Explique.

   Resposta: 30. No TCP o próximo número de seqüência do segmento é definido pelo número de seqüência do segmento anterior, acrescido da quantidade de *bytes* de dados enviados no segmento anterior (observado o limite máximo do número de seqüência). Neste caso, o número de seqüência 140 é definido pelo número de seqüência 110 somado ao número de *bytes* de dados transmitidos nesse primeiro segmento, que no caso é 30.

   b. Suponha que o primeiro segmento foi perdido mas o segundo foi entregue a B. Na confirmação que B envia para A, qual o valor contido no campo ACK do segmento? Explique.

   Resposta: 110. Isto no caso em que o segmento anterior ao segmento com número de seqüência 110 já tenha sido confirmado. O TCP sempre envia no campo de confirmação o próximo *byte* em seqüência (isto é, em ordem), por ele esperado.

6. **(1,5 pontos cada item)** Sabemos que o TCP realiza controle de fluxo e controle de congestionamento. Responda:

   a. Qual o objetivo do controle de fluxo? De que forma é realizado o controle de fluxo?

   Resposta: O controle de fluxo serve para evitar que o transmissor da conexão TCP envie mais dados que a capacidade de recepção do receptor dessa conexão TCP. Para tal, o transmissor precisa conhecer o espaço disponível no *buffer* de recepção do lado receptor da comunicação. Este dado é fornecido através do campo "Receive Window", que está presente no cabeçalho do TCP e que indica o espaço, em *bytes* , disponível no *buffer* de recepção. Como o TCP é *fullduplex,* o campo "Receive Window" é preenchido, no lado receptor da conexão, todas as vezes que um segmento é enviado para o lado transmissor. Com o mecanismo de controle de fluxo, um problema poderia ser ocasionado quando a janela de recepção disponível chegasse a zero. Nessa situação, o transmissor não enviaria dados para o receptor e caso o receptor não tivesse dados para enviar na conexão, o transmissor não teria como saber que o espaço na janela de recepção foi liberado. O TCP resolve este problema forçando o envio periódico de pacotes, por parte do transmissor, com apenas um *byte* de dados, quando a capacidade da janela de recepção chega a zero.

   b. Qual o objetivo do controle de congestionamento? De que forma é realizado o controle de congestionamento?

   Resposta: O controle de congestionamento tem como alvo a infra-estrutura de comunicação da Internet, que interliga os dois hospedeiros, e tem como objetivo evitar um colapso de comunicação no interior da rede IP. O mecanismo de controle de congestionamento no TCP, que é baseado no "Aumento Aditivo, Diminuição Multiplicativa" (AIMD), mantém a conexão TCP em dois estados:

   - Inicio lento (*slow start* – SS): A conexão TCP está neste estado quando a janela de congestionamento for inferior ao limiar (*threshold)*. A cada confirmação (ACK) recebida em seqüência o tamanho da janela de congestionamento é acrescido do tamanho de um MSS *(Maximum Segment Size),* o que resulta na duplicação da janela de congestionamento a cada RTT

(*Round Trip Time*).

- Prevenção de congestionamento (c*ongestion avoidance* - CA): A conexão TCP está neste estado quando a janela de congestionamento (CongWin) for igual ou superior ao limiar (*threshold*). A cada confirmação (ACK) recebida em seqüência o tamanho da janela de congestionamento é acrescido através da fórmula:

$$CongWin = CongWin + MSS \times MSS/CongWin$$

A aplicação desta fórmula resulta no acréscimo do tamanho de um MSS ao tamanho da janela de congestionamento a cada RTT (*Round Trip Time*). Obviamente, este crescimento na janela de congestionamento é efetuado de forma que não infrinja controle de fluxo.
Outros eventos, além da recepção de um ACK esperado e em seqüência, são utilizados neste mecanismo:

- Quando um ACK é recebido em duplicata é incrementado o contador de "ACKs em duplicata" e quando o terceiro ACK em duplicata é recebido à janela de congestionamento é reduzida à metade e o limiar (*threshold*) também recebe este mesmo valor. Portanto, a conexão entra no estado de prevenção do congestionamento.
- Quando o temporizador expira (*timeout*) o limiar recebe o valor da metade do tamanho da janela de congestionamento e a janela de congestionamento é reduzida para seu tamanho mínimo, ou seja, um MSS. Portanto, a conexão TCP entra na fase de inicio lento (*Slow Start*).