Fundação CECIERJ - **Vice Presidência de Educação Superior a Distância**

Curso de Tecnologia em Sistemas de Computação
Disciplina: Redes de Computadores I
AD2 - 2° semestre de 2012.
GABARITO

---

1. **(1,5 pontos)** Considere o protocolo Retorne a N (*Go Back N*) para transferência confiável de dados. Suponha que o espaço de números de sequência usado é de tamanho k (isto é, vai de 0 até k-1). Responda: qual o maior tamanho de janela (do transmissor) que pode ser usado, para garantir que o protocolo não falhe? Explique sua resposta.

   Resposta:

   No protocolo Retorne a N o tamanho da janela do receptor tem por definição tamanho **1** (isto é, o receptor só recebe o segmento em sequência). Considerando o espaço dos números de sequência é de **k** diferentes números (isto é, os números vão de **0** até **k-1**), o tamanho máximo da janela do transmissor tem que ser no máximo igual a **k-1**. Suponha por exemplo, **k=8** e portanto os números de sequências possíveis são **0, 1, ..., 7**. Isto nos levaria a imaginar que o tamanho da janela do transmissor pode ser **8** (isto é, o transmissor pode enviar **8** segmentos, numerados de **0** até **7**, sem ter recebido qualquer confirmação). Imagine então, que o transmissor tem janela de tamanho **8** e que ele envia os **8** segmentos, e recebe a confirmação para cada um deles individualmente ou cumulativamente para todos eles. Tendo recebido essas confirmações, o transmissor avança sua janela, podendo enviar oito novos segmentos reutilizando os números de sequência de **0** até **7**. Imagine então que esses oito novos segmentos são perdidos e que o receptor envia uma duplicata de confirmação para o último segmento recebido corretamente em sequência, isto é, o receptor confirma mais uma vez o segmento **7** da primeira leva de segmentos. O transmissor não tem como distinguir que trata-se de uma duplicata da confirmação e considera-a como sendo a confirmação que está aguardando para a nova leva de **8** segmentos enviados e não confirmados, descartando então os oito segmentos que não foram entregues ao receptor. Então o protocolo falha (o protocolo com tamanho de janela igual a **8**, não faz transferência de dados confiável)! Desse modo, o tamanho da janela do transmissor pode ser no máximo 7, isto é, **k-1. Sugestão: repita o raciocínio considerando a janela do transmissor de tamanho 7, e certifique-se que assim o protocolo não falha!**

2. **(1,5 pontos)** Considere o protocolo Repetição Seletiva (*Selective Repeat*) para transferência de dados confiável. Considere ainda que a faixa de números de seqüência é: 0, 1, 2, ..., k-1. Qual o maior tamanho de janela possível para que o protocolo não falhe?

Resposta:

Tamanho máximo da janela para que o protocolo não falhe é igual a **k / 2**. Como na Questão 7, Suponha por exemplo, k=8 e portanto os números de seqüências possíveis são 0, 1, ..., 7. Admita os tamanho das janelas do transmissor e do receptor igual a 7 (podemos tentar esse tamanho de janela já, que esse valor nos garante que o protocolo Retorne a N não falha). Nesse caso, na janela do receptor são aguardados os segmentos 0, 1. 2, ...,6. Imagine então, que o remetente envia os segmentos 0, 1, 2, ..., 6 e todos eles são recebidos no destino e confirmados, passando o receptor a aguardar sete novos segmentos, os segmentos 7, 0, 1, ..., 5. Porém por uma infeliz coincidência, todas as confirmações enviadas são perdidas! Após a temporização o remetente re- envia o segmento 0, que chega ao destino e o receptor recebe essa cópia como sendo um novo dado. Então o protocolo falha (o protocolo com tamanho de janela igual a 7, não faz transferência de dados confiável)! Para o protocolo funcionar corretamente, o tamanho da janela pode ser no máximo 4, isto é, **k / 2.** Na verdade, para que o protocolo não falhe, o conjunto dos números de seqüência de uma janela e a da janela seguinte, devem ter intercessão vazia. **Sugestão: repita o raciocínio considerando a janela do transmissor de tamanho 6 e 5 e certifique -se que nos dois casos o protocolo continua falhando. Finalmente repita o raciocínio considerando a janela do transmissor de tamanho 4 e verifique que desse modo, o protocolo não falha!**

3. **(1,5 pontos)** O hospedeiro A está enviando um arquivo enorme para o hospedeiro B através de uma conexão TCP. Nessa conexão não há perda de pacotes e os temporizadores nunca se esgotam. Seja R bps a taxa de transmissão do enlace que liga o hospedeiro A à Internet. Suponha que o processo que executa no hospedeiro A é capaz de enviar dados para o seu *socket* TCP à uma taxa de S bps, onde S = 10 x R. Suponha ainda que o *buffer* de recepção do TCP (no hospedeiro B) seja grande o suficiente para conter o arquivo inteiro, e que o *buffer* de envio TCP (no hospedeiro A) pode conter apenas um por cento do arquivo. O que impediria o processo que executa no hospedeiro A de passar dados continuamente para o seu *socket* TCP a taxa de S bps: o controle de fluxo TCP; o controle de congestionamento TCP; ou alguma outra coisa? Explique sua resposta.

   Resposta:

   Neste problema não existe o perigo do transmissor sobrecarregar (inundar) o receptor, pois o *buffer* do receptor é capaz de armazenar o arquivo inteiro, logo o controle de fluxo não impede o **hospedeiro A** de passar dados continuamente para o seu *socket* TCP a taxa de **S** bps. Considerando que não existem perdas e que os ACKs retornam antes dos temporizadores expirarem, o mecanismo de controle do congestionamento não influi na taxa do transmissor. Contudo, o processo do **hospedeiro A** não poderá passar dados continuamente para o *socket,* pois o *buffer* de transmissão é pequeno e logo fica cheio. Portanto, tão logo que o *buffer* de transmissão está cheio, o processo repassa os dados à uma taxa média R que é muito menor que S.

4. **(1,0 ponto)** Uma confirmação TCP perdida não necessariamente força uma retransmissão. Por que?
   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. **(1,5 pontos)** Os hospedeiros A e B estão conectados diretamente por um enlace de 200 Mbps. Existe uma conexão TCP entre esses dois hospedeiros, e o hospedeiro A está enviando um grande arquivo para o hospedeiro B através dessa conexão. O hospedeiro A pode enviar dados da aplicação a taxa de 100Mbps porém no hospedeiro B o *buffer* de recepção TCP (*receive buffer*) é lido pela aplicação a taxa de 50Mbps. Descreva o efeito do controle de fluxo do TCP nesse cenário.
   Resposta:
   O hospedeiro A envia dados para o *buffer* do receptor mais rapidamente do que o receptor do hospedeiro B pode retirar dados do *buffer*. O *buffer* de recepção em B rapidamente fica cheio devido a maior velocidade na entrada de dados que na retirada dos mesmos. Quando o *buffer* estiver cheio, o hospedeiro B sinaliza para A que pare de enviar dados atribuindo à janela de recepção o valor zero (RecWindow=0). O hospedeiro A para de enviar dados até que receba um valor para a janela de recepção superior a zero (RecWindow>0). Desta forma, o hospedeiro A pára e inicia a transmissão de dados como uma função dos valores recebidos para a janela de recepção do hospedeiro B. Na média, ao longo do tempo, a taxa de transmissão na qual o hospedeiro A envia dados para o hospedeiro B, durante a conexão não será superior a 50Mbps, evitando que o lado transmissor suplante a capacidade de recepção do outro lado da conexão.

6. **(1,5 pontos)** Para que serve o controle de fluxo realizado na camada de transporte da Internet? Explique como esse serviço é implementado.
   Resposta:
   O controle de fluxo visa impedir que um receptor fique sobrecarregado por estar recebendo pacotes a uma taxa superior à que este possa consumi-los.
   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 *full duplex,* 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.

7. **(1,5 pontos)** Para que serve o controle de congestionamento realizado na camada de transporte da Internet? De que forma é realizado o controle de congestionamento?

Resposta:

O controle de congestionamento visa proteger a rede de uma carga de pacotes superior a sua capacidade. Em outras palavras, 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, 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, de um MSS. Portanto, a conexão TCP entra na fase de inicio lento (*Slow Start*).