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 2007.
Gabarito

1. Explique para que serve o processo de demultiplexação realizado na camada de transporte. (1,0 ponto)

Resposta:
Um processo (como parte de uma aplicação de rede) pode ter um ou mais *sockets*, portas pelas quais dados passam da rede para o processo e do processo para a rede. A tarefa de entregar dados contidos em um segmento da camada de transporte para a porta correta é denominada demultiplexação. Para tanto, na extremidade receptora, a camada de transporte examina os campos do cabeçalho do segmento para identificar a porta receptora e direciona o segmento para o *socket* correto.

2. Suponha um processo que executa no hospedeiro C tem um *socket* com número de porta 6789. Suponha que dois hospedeiros A e B, enviem segmentos UDP para a porta de destino 6789 do hospedeiro C. Responda:

    a) Ambos os segmentos serão direcionados para o mesmo *socket* no hospedeiro C? Se sua resposta é sim, como o processo que executa no hospedeiro C, sabe que esses dois segmentos têm origem em dois hospedeiros diferentes? (1,0 ponto)

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.

3. Suponha que um servidor Web executa no hospedeiro C e escuta na porta 80. Suponha que esse servidor Web usa conexões persistentes e que está nesse momento, recebendo requisições de dois hospedeiros A e B diferentes. Explique:

    a) Essas requisições são direcionadas para o mesmo *socket* no hospedeiro C? (0,5 pontos)

    Resposta:
    Não. Para cada conexão persistente, o servidor de Web cria um *socket* de conexão separado. Cada *socket* de conexão é identificado pela 4-tupla <endereço IP de origem, número de porta de origem, endereço IP de destino, número de porta de destino >. Quando hospedeiro C recebe um datagrama IP, ele examina esses quatro campos nos cabeçalhos das camadas de rede e de transporte (datagrama/segmento) para determinar para qual *socket* deve passar a carga útil (*payload*) do segmento TCP.

b) Se as requisições são direcionadas para *sockets* diferentes, podem esses diferentes *sockets* ter o mesmo número de porta 80? (1,0 ponto)

Resposta:
Como vimos no item (a), cada *socket* de conexão é identificado por quatro informações: <(IP origem, número da porta de origem, IP destino e número da porta destino>. Quando o hospedeiro C recebe um datagrama IP, este examina estes quatro campos no datagrama/segmento para determinar para qual *socket* o conteúdo de informação do segmento TCP deve ser repassado. Assim, as requisições de A e B vão para *sockets* diferentes. O identificador para ambos *sockets* tem 80 como porta de destino, porém, os identificadores para estes *sockets* têm valores diferentes para o endereço IP de origem. Diferentemente do que ocorre no UDP, quando a camada de transporte repassa o conteúdo de informações de um segmento de TCP para o processo da aplicação, não especifica o IP de origem, pois isto é implicitamente especificado pelo identificador do *socket*.

4. Descreva por que um desenvolvedor de uma aplicação distribuída escolhe executar sua aplicação sobre UDP em vez de executá-la sobre TCP. (1,0 ponto)

Resposta:
A escolha do UDP como o protocolo da camada de transporte é baseada em:

- **A aplicação não necessita da transferência de dados confiável:** Algumas aplicações suportam que certo percentual de dados sejam perdidos e portanto o uso do UDP é indicado.
- **Taxa de envio não regulada**: O desenvolvedor da aplicação não quer sua aplicação fique sujeita ao controle de congestionamento realizado pelo TCP que pode, em tempo de congestionamento, reduzir a taxa de transmissão da aplicação. No TCP existem os mecanismos de controle de congestionamento e de fluxo que regulam a taxa de envio de segmentos, sem a interferência da aplicação transmissora. Este controle na taxa de envio de segmentos pode ser crítica para aplicações que requisitam uma taxa de transmissão mínima e que são tolerantes a perda de pacotes até um certo nível.
- **Não há estabelecimento de conexão:** No TCP existe o *three way handshake* antes que a transferência de dados seja iniciada.
- **Não há estado de conexão:** O TCP mantém o estado de conexão nos sistemas finais de origem e destino. Esse estado inclui *buffers* de envio e recepção, parâmetros de controle de congestionamento e parâmetros numéricos de seqüência e de reconhecimento, desta forma esta manutenção do estado tem custos de processamento e espaço alocado.
- **Pequeno overhead no cabeçalho do pacote:** O segmento TCP tem 20 bytes de cabeçalho, enquanto o do UDP tem somente 8 bytes.

5. Considere o protocolo Retorne a N (*Go Back N*) 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?

(1,0 ponto)

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 seqüência). Considerando o espaço dos números de seqüência 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 seqüências 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 seqüência 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 seqüência, 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!**

6. 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?

(1,0 ponto)

Resposta:

Tamanho máximo da janela para que o protocolo não falhe é igual a **k / 2**. Como na Questão 5, 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!**

7. Responda verdadeiro ou falso, explicando sua escolha: (0,5 ponto)

   a) Imagine que o hospedeiro A envie ao hospedeiro B, por uma conexão TCP, um segmento contendo 16 bytes de dados e com número de seqüência 60. Nesse mesmo segmento, o número contido no campo de confirmação é obrigatoriamente 76.

Resposta:
Falso. Os valores iniciais para os números de seqüência para cada um dos lados da conexão TCP são definidos durante o estabelecimento da conexão TCP (*three way handshake*), e cada uma das partes faz a escolha independentemente. Portanto o campo de confirmação, ao longo a transmissão na conexão TCP, reflete essa escolha inicial.

8. Para que serve o campo "Janela de Recepção" (ou "RcvWindow") no cabeçalho do segmento TCP? (1,0 ponto)

Resposta:
O campo "RcvWindow" indica a quantidade de *bytes* disponível no "*buffer* de recepção" no lado receptor de uma conexão TCP. O conteúdo desse campo é usado para evitar que o transmissor envie mais dados do que o receptor é capaz de receber, evitando assim o descarte de dados, por falta de espaço de armazenamento no lado receptor de uma conexão TCP.

9. Como é escolhido o valor do temporizador de uma conexão TCP? (1,0 ponto)

Resposta:
Denominamos "RTT amostra" (*Round Trip Time* amostra), o tempo transcorrido desde a entrega do segmento para o IP até o momento em que é recebida a sua confirmação. A maioria das implementações de TCP faz uma única medição "RTT amostra" por vez (isto é, para apenas um dos segmentos transmitidos e ainda não confirmados). As medidas obtidas para os "RTT amostra" de diferentes segmentos, variam devido a congestionamento nos roteadores e a variação de carga nos hospedeiros. Por causa dessa variação qualquer valor medido para "RTT amostra" pode ser atípico. Portanto para estimar um valor para o RTT típico, é preciso determinar algum tipo de média sobre os valores "RTT amostra" obtidos. Para tal. o TCP mantém uma média, denominada "RTT estimado" dos valores "RTT amostra". Essa média é atualizada sempre que um novo valor de "RTT amostra" é obtido, da seguinte forma:

$$\mathbf{RTT\_estimado = (1 - a) * RTT\_estimado + a * RTT\_amostra}$$

Este cálculo fornece a média exponencial ponderada, na qual a influência de cada "RTT_amostra" diminui exponencialmente com o tempo. Um valor típico usado para a é 0,125.
Para determinar o valor da temporização de uma conexão TCP, ao "RTT_estimado" é adicionada uma margem de segurança. Esta margem de segurança é calculada através do desvio da amostra (denominado "desvio RTT") em relação ao "RTT estimado". Assim "desvio RTT" é determinado da seguinte forma:

$$\mathbf{desvio\_RTT = (1 - ß) * desvio\_RTT + ß * | RTT\_amostra - RTT\_estimado |}$$

O valor recomendado para ß é 0,25. Com os valores de "RTT estimado" e de "desvio RTT", o valor do temporizador é calculado da seguinte forma:

$$\mathbf{Temporizador = RTT\_estimado + 4 * desvio\_RTT}$$

10. Discorra sobre o mecanismo “Aumento Aditivo, Diminuição Multiplicativa” (*Additive-Increase, Multiplicative-Decrease* - AIMD) usado no controle de congestionamento realizado pelo TCP. (1,0 ponto)

Resposta:

A idéia usado no controle de congestionamento do TCP é que o remetente reduza sua taxa de transmissão (isto é, reduzindo o tamanho de sua janela) quando ocorre um evento de perda. A abordagem da “diminuição multiplicativa”, reduz “a metade o valor da janela do remetente após um evento de perda de segmento. Esse processo se repete sempre um evento de perda é detectado, até o valor mínimo de 1 MSS (*Maximum Segment Size*).

O) TCP sua taxa de envio quando percebe que não há congestionamento, isto é, quando chegam confirmações para dados que ainda não tinham sido confirmados anteriormente. O TCP então aumenta sua taxa de envio lentamente, aumentando sua janela de 1 MSS sempre que recebe uma nova confirmação. Em resumo o TCP aumenta sua taxa de envio aditivamente quando percebe que o caminho fim-a-fim não está congestionado e a reduz multiplicativamente quando detecta (por meio de um evento de perda) que o caminho está congestionado.