Redes de Computadores

TP2: transmissão confiável

Prática com elementos de aplicação, utilizando select() e temporizações

Trabalho individual ou em duplas

Valor: 18 pontos.

Data de entrega: consulte o calendário do curso Versão PDF deste documento

Última alteração: 6. Abril 2014

O problema

Neste trabalho o objetivo é implementar um serviço de transmissão confiável utilizando UDP como protocolo de transporte. Como esse protocolo não oferece um serviço confiável, será função do seu trabalho implementar um protocolo de janela deslizante para garantir entrega confiável e em ordem. Soluções sem janela deslizante (só *stop-and-wait*)serão aceitas, mas avaliadas com uma penalização de 25%.

O programa cliente enviará o conteúdo de um arquivo passado como parâmetro para o servidor. O servidor receberá o arquivo e o armazenará no disco.

Será necessário executar experimentos para fins de avaliação do desempenho do protocolo, com e sem falhas. O número de pacotes retransmitidos também deve ser computado para fins de comparação.

Para fins de avaliação utilizaremos a topologia /custom/topo-2sw-2host.py do mininet, como no TP1, com a configuração parecida (mas não igual) à usada naquele trabalho: os links dos hosts terão 1 Mbps de banda e um atraso de 10 ms, enquanto o link entre switches terá uma banda menor (100 Kbps) e atrasos variando de 10 ms a 500 ms. Seu protocolo deve ser avaliado com diferentes tamanhos de janelas, mantendo-se um atraso de 100 ms no link intermediário. Os valores adequados de variação do tamamnho da janela devem ser encontrados pelos implementadores dos protocolos em cada caso. Para valores menores, pode ser recomendável reduzir o tamanho do arquivo para evitar esperas muito longas.

Além de mostrar o *throughput* oferecido pelo protocolo, devem também ser apresentados os resultados de execução quando o link entre os *switches* tiver uma taxa de perda de 5% (loss=5).

Cada grupo deverá apresentar um relatório descrevendo o funcionamento do seu protocolo e comparando os mecanismos de transmissão no envio de um arquivo de aproximadamente 1 MB.

Definição do protocolo

Neste trabalho, como não teremos TCP para garantir a ordem, vocês serão responsáveis por criar um protocolo confiável sobre UDP usando janela deslizante. Para isso, vocês devem definir quais serão os campos do cabeçalho do seu pacote (que será enviado dentro do pacote UDP), que encapsulará os dados do arquivo.

O protocolo nesse caso será unidirecional, isto é, ele deve ser construído para enviar dados apenas na direção do servidor para o cliente. Mensagens de confirmação podem ser necessárias em diferentes momentos e uma técnica de detecção de erros também deverá ser utilizada, pois erros poderão ocorrer durante os testes.

Recomendo que os cabeçalhos sejam de tamanho fixo e representados em binário, por ser mais simples e eficiente que usar strings para isso. A melhor forma de lidar com isso é definir os cabeçalhos como um array de bytes em Python e prencher esse array como for necessário.

Um grande desafio deste trabalho é o tratamento de temporizações. Isso pode ser feito de diferentes maneiras, usando uma temporização associada à recepção, usando *timers* em Python ou usando a função *select*.

Atenção: apesar de Python ser interpretada, pode haver problemas na implementação de detalhes de *timers*, *threads* e *select* no Windows. Para evitar surpresas, recomenda-se que todos os testes sejam feitos no Linux ou no Mac OS X.

Detalhes de operação

As mensagens UDP não devem ultrapassar o MTU da rede, isto é, cada mensagem UDP não deve transportar mais que 1460 bytes de dados (isso, mas 20 bytes do cabeçalho UDP e outros 20 do cabeçalho IP completarão os 1500 bytes de dados em um quadro Ethernet.

O cliente deve apresentar as seguintes opções de linha de comando:

-f arquivo:
: especifica em **arquivo** o nome do arquivo a ser transferido.

-h endereço:
: especifica em **endereço** o nome ou endereço IP do servidor.

-p porto:
: especifica em **porto** o número do porto no qual o servidor está aguardando conexões.

-w:
: especifica o uso de stop-and-wait; não pode ser usado em conjunto com **-g** ou **-a**.

-g:
: especifica o uso de janela deslizante com go-back-n; não pode ser usado em conjunto com **-w** ou **-a**.

-a:
: especifica o uso de janela deslizante com selective ack; não pode ser usado em conjunto com **-w** ou **-g**.

-t taxa:
: especifica em **taxa** a taxa (percentual) de perda de pacotes enviados para o cliente (valores possíveis: número de ponto flutuante -- double -- indicando a porcentagem de pacotes ``perdidos'', em média).

O servidor deve apresentar as seguintes opções de linha de comando:

-o saída:
especifica em **saída** o nome com o qual o arquivo recebido deve ser armazenado.
-p porto:
especifica em **porto** o número do porto no qual o servidor está aguardará por conexões.
-t taxa:
especifica em **taxa** a taxa de perda de pacotes enviados para o servidor.

O protocolo utilizado deve ser projetado por cada grupo, não havendo quaisquer restrições além das mencionadas acima. O relatório a ser apresentado deve conter detalhes como o formato dos cabeçalhos das mensagens, o processo de transmissão do arquivo e de identificação do início e fim da conexão, determinação dos intervalos de temporização, etc.

Detalhes de implementação

Para implementar cada programa, deve-se isolar o processo de ler o arquivo do disco da implementação do protocolo de transmissão propriamente dito. Para isso, a forma mais fácil é utilizar *threads*: no transmissor, uma *thread* executa um *loop*, lendo o arquivo do disco e colocando-o em um buffer (circular), que é então acessado por uma ou mais threads que implementam o protocolo; no receptor, o inverso acontece: uma thread lê de um buffer circular preenchido pelo protocolo para então escrever no arquivo. Obviamente, o acesso ao buffer circular deve ser coordenado e deve-se prever as situações onde o buffer pode estar cheio ou vazio para as duas partes. No caso do protocolo, isso tem impacto sobre o controle de fluxo no protocolo de janela deslizante -- que é diferente do processo de avanço da janela.

Informações úteis:

As métricas de comparação entre os mecanismos de transmissão devem ser: tempo total de transmissão e número de pacotes retransmitidos.

O principal desafio deste trabalho é o uso de temporizações. Estas podem ser obtidas de duas formas: através do uso do parâmetro de temporização do select ou através de um tratador de sinais (signal handler) instalado em um sinal de temporização que interrompa o programa quando necessário. Por exemplo, você pode usar a primitiva ualarm. Alguns bons tutoriais sobre o uso de sinais estão disponíveis na rede; por exemplo, http://docs.hp.com/en/32650-90372/ch02s14.html e http://users.actcom.co.il/~choo/lupg/tutorials/signals/signals-programming.html. Note que qualquer chamada do sistema que seja considerada ``lenta'' (blocante) pode retornar com um código de erro (EINTR) se foi interrompida por um sinal.

Uma possível maneira de se implementar a simulação de uma rede 95% confiável seria sortear um número utilizando a função random() após toda chamada pela função recv() (isto se aplica tanto para o servidor quanto para o cliente). Em 5% dos

números sorteados, basta ignorar os dados lidos via recv(). O mesmo se aplica caso o select seja usado para receber as mensagens: toda vez que ele retorne com uma mensagem válida deve-se sortear se ela vai mesmo ser usada ou se ``na verdade'' ela se perdeu na rede.

Pretendemos disponibilizar um programa extra que simulará um canal de banda e atrasos pré-definidos para ser usado durante os testes. Mais detalhes sobre isso serão fornecidos oportunamente.

Uma operação que pode ser útil dependendo da forma de implementação do cliente transmissor é uma função para determinar o tamanho de um arquivo no disco. Essa informação pode ser obtida através da função stat() no Unix/Linux. Por simplicidade a função a seguir pode ser usada para simplificar o programa:

```
#include <sys/types.h>
#include <sys/stat.h>
#include <unistd.h>

int
fsize( char* fname ) /* retorna tamanho (bytes) do arquivo fname*/
{
   struct stat stat_info;
   return (stat(fname,&stat_info)<0)? -1: (int) stat_info.st_size;
}
```

Utilize um arquivo de tamanho adequado nos experimentos; esse tamanho deve ser tal para levar a uma transmissão de aproximadamente um minuto na banda disponível (banda da rede local ou do programa simulador de canal, se ele estiver disponível). Certifique-se de gerar a assinatura md5 do arquivo original e do arquivo escrito pelo receptor para se certificar de que seu protocolo funciona corretamente.

Submissão eletrônica

O código fonte dos programas cliente e servidor e o relatório devem ser empacotados (.zip ou .tar.gz) e entregues pelo moodle.

Critérios de avaliação

É esperado que o relatório apresentado pelo grupo contenha:

- breve descrição do problema;
- descrição do protocolo implementado (funcionamento, formato de pacotes, etc.);
- decisões de implementação;
- descrição do experimento comparativo realizado;
- discussão dos resultados do experimento;
- conclusão.

Serão atribuídos pontos também para:

•execução correta do programa;
•execução completa do programa (implementação de todos os mecanismos);
•apresentação e organização do documento/relatório;
•domínio do problema exibido pelos alunos.

Pode vir a ser realizada uma entrevista com cada grupo. Se isso acontecer, cada componente do grupo será avaliado separadamente durante a entrevista.

Observações Gerais

1.Programas cujo comportamento não siga o formato descrito acima ou que sejam submetidos de forma incorreta (especialmente no que diz respeito à submissão eletrônica) serão penalizados no processo de avaliação.
2.Este não é um trabalho fácil, apesar do código ser relativamente curto. Em particular, o tratamento de temporizações ou uso do select são elementos sujeitos a erros em situações sutis. Comece com programas de teste simples onde vocês garantam que esses elementos sejam dominados.
3.**Dúvidas:** não hesitem em publicar suas dúvidas no moodle, mas lembrem-se que **não é permitido postar trechos de código**.
Tentaremos responder toda pergunta em menos de 48 horas.
4.Comece a fazer este trabalho logo, enquanto o problema está fresco na memória e o prazo para terminá-lo está tão longe quanto jamais poderá estar.
5.Vão valer pontos clareza, completude e apresentação da documentação.
6.Como discutido no código de conduta da disciplina, o trabalho deve ser desenvolvido de forma independente por cada grupo. É permitido discutir os aspectos do programa e soluções de forma abstrata, mas é terminantemente proibido compartilhar programas ou trecho de programas. Tal comportamento será punido severamente.