



Universidade Federal de Minas Gerais
Departamento de Ciência da Computação



Redes de Computadores 2011/1

Redes de Computadores

TP1: *sockets*, medição de desempenho
Prática para exercitar o domínio do uso de C e sockets

Trabalho individual
Data de entrega: 27/04/2011

O problema

Implementar um par de programas que operem no modelo cliente-servidor e exercitem tanto a transmissão unidirecional quanto a comunicação do tipo requisição-resposta sobre o protocolo TCP. A implementação deve utilizar a biblioteca de sockets do Unix (Linux).

Operação

O processo de comunicação entre cliente e servidor deverá seguir um padrão simples, ilustrado a seguir:

Cliente:

processa argumentos da linha de comando:

host_do_servidor porto_servidor nome_arquivo tam_buffer

chama gettimeofday para tempo inicial

faz abertura ativa a host_do_servidor : porto_servidor

envia string com nome do arquivo (terminada em zero)

abre arquivo que vai ser gravado - pode ser fopen(nome, "w+")

loop recv buffer até que receba zero bytes ou valor negativo

escreve bytes do buffer no arquivo (fwrite)

atualiza contagem de bytes recebidos

fim_loop

fecha conexão e arquivo

chama gettimeofday para tempo final e calcula tempo gasto

imprime resultado:

"Buffer = %5u byte(s), %10.2f kbps (%u bytes em %3u.%06u s)

fim_cliente.

Servidor:

processa argumentos da linha de comando:

porto_servidor tam_buffer

faz abertura passiva e aguarda conexão

recebe, byte a byte até o zero, o string com nome do arquivo

abre arquivo que vai ser lido -- pode ser fopen(nome, "r")

se deu erro, fecha conexão e termina

loop lê o arquivo, um buffer por vez até fread retornar zero

envia o buffer lido
se quiser, contabiliza bytes enviados
fim_loop
fecha conexão e arquivo
chama `gettimeofday` para tempo final e calcula tempo gasto
se quiser, imprime nome arquivo e no. de bytes enviados
fim_servidor.

De forma resumida, o cliente deve se conectar ao servidor, enviar um string com o nome do arquivo desejado, receber o arquivo um buffer de cada vez e salvar os dados no disco à medida que eles chegam. Quando não houver mais bytes para ler o cliente fecha a conexão e o arquivo e gera uma linha com os dados da execução. O servidor por sua vez deve operar de forma complementar.

Medições de desempenho

Uma vez que os programas estejam funcionando corretamente, deve-se desenvolver uma avaliação do desempenho do par de programas. Deve-se medir o *throughput* da comunicação, isto é, a taxa de transferência obtida entre cliente e servidor (basicamente, o número total de bytes enviados dividido pelo tempo medido no cliente).

As medições devem verificar como o desempenho varia quando diferentes tamanhos de buffer são utilizados. Em particular, deve-se incluir nas medições os casos com mensagens de tamanho 2^i bytes, $0 \leq i \leq 16$. Outros valores podem ser escolhidos conforme suas observações indicarem a necessidade.

O tamanho do arquivo pode ser escolhido de forma a garantir um tempo de teste nem muito longo nem muito curto. Para testes em que o par de programas executa na mesma máquina, um arquivo de aproximadamente 3 MB pode ser um bom ponto de partida.

O relatório

Junto com os programas desenvolvidos deve ser entregue um relatório (PDF) contendo as seguintes seções:

1. Introdução: descrição do objetivo do trabalho
2. Metodologia: dados sobre os experimentos, como a configuração das máquinas utilizadas, a localização das mesmas na rede. Indique também como foram feitas as medições (`gettimeofday`, `imaging`), quantas vezes o teste foi executado, se foram execuções diferentes do programa ou apenas um loop ao redor do programa todo para fazer tudo um certo número de vezes.
3. Resultados: apresente a informação coletada, tanto na forma de tabelas quando na forma de gráficos.

Cada tabela deve conter, para o experimento em questão, uma linha para cada tamanho de mensagem, com o número de mensagens enviadas, o tempo total médio medido, o desvio padrão dos tempos medidos e a banda média observada no experimento.

Cada gráfico deve usar escalas adequadas (logarítmicas ou lineares, conforme o caso), os eixos devem ser identificados e possuir claramente a identificação das unidades em cada um. Os resultados devem ser apresentados por linhas retas interligando os pontos medidos, que devem ser destacados com marcas claras. Se um gráfico contiver mais que um conjunto de pontos as linhas devem ser claramente identificadas. Se possível, cada gráfico deve incluir barras verticais indicando a variância de cada valor calculado.

4. Análise: para cada experimento, discuta os resultados observados. Os resultados foram de acordo com o esperado? Você é capaz de explicar por que as curvas se comportam como o fazem? Houve algum elemento claramente de destaque nos resultados que merece uma análise especial (por exemplo, um pico/vale inesperado nos gráficos, desvios muito significativos nas medições)?
5. Conclusão: como todo trabalho técnico, algumas palavras finais sobre o resultado do trabalho, tanto das observações quanto do seu aprendizado sobre o assunto.

O relatório não precisa incluir a listagem dos programas.

Submissão eletrônica

A entrega eletrônica deve ser feita para **submission.vieira@gmail.com** e deve constar de um arquivo do tipo zip ou tar.gz contendo os seguintes documentos:

- todos os arquivos **fonte** utilizados para construir os programas (arquivos de terminação .c/.cpp, .h, makefile);
- o programa cliente chamará: clienteFTP
- o programa servidor chamará: servidorFTP
- uma cópia eletrônica do relatório;
- se possível, arquivos contendo os dados coletados, seja como planilhas ou como arquivos texto contendo os dados na forma gerada pelos programas.

Não inclua nesse conjunto os arquivos objeto (.o) nem os executáveis utilizados!

Observações Gerais

1. **Programas cuja saída não sigam o formato descrito acima, ou que exijam dados de entrada em formato diferente do descrito, ou que sejam submetidos de forma incorreta (especialmente no que diz respeito à submissão eletrônica) serão penalizados no processo de avaliação.**
2. **Dúvidas:** não hesitem em escrever para o professor e o monitor . Tentaremos responder toda pergunta em menos de 48 horas.
3. 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.
4. **Vão valer pontos clareza, indentação e comentários no programa.**

Última alteração: Abril 5, 2011

Versão HTML produzida por Marcos Vieira