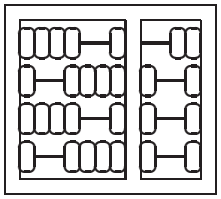
UNIVERSIDADE ESTADUAL DE CAMPINAS

INSTITUTO DE COMPUTAÇÃO



SERVIDOR DE AGENDA BASEADO EM SOCKET TCP

Relatório do primeiro laboratório de MC823

Aluno: Marcelo Keith Matsumoto **RA**: 085937

Aluno: Tiago Chedraoui Silva **RA**: 082941

Resumo

O TCP (Transmission Control Protocol) é um protocolo do nível da camada de transporte [4]. Algumas de suas características são transferência garantida, controle de congestão, ser orientado à conexão e controle de fluxo.

Utilizando esse protocolo desenvolveu-se uma aplicação distribuída que simulava uma agenda para o mês de abril,com diversas funções (inserção de dados na agenda, recuperação de dados, remoção de dados), através da qual foi possível observar o funcionamento de uma comunicação via TCP entre um cliente e um servidor em máquinas diferentes. Para isso, a interação entre ambos ocorria através do envio de dados do cliente para o servidor, assim como na outra direção, utilizando sockets criados.

Por fim, foi realizado uma análise dos tempo de de comunicação, processamento e total do programa, no qual concluiu-se que os tempos de processamento são significativamente maiores que os tempos de conexão.

Sumário

1 Motivação

Atualmente, com o crescente aumento de dispositivos móveis e computadores conectados à rede, o conhecimento da comunicação tem se tornado cada vez mais importante para quem trabalha na área de tecnologia. Um dos maiores exemplos da importância da comunicação entre computadores é em um sistema distribuído.

Um sistema distribuído é uma "coleção de computadores independentes que se apresenta ao usuário como um sistema único e consistente"[1]. Para que isso seja possível, diversos computadores estão interligados por uma rede de computadores, através da qual compartilham entre si objetos (como arquivos, informações, processamento, etc) e são responsáveis por manter uma consistência nesses objetos. Portanto, hoje, um servidor não é apenas um computador, mas sim vários computadores em locais diferentes que aparentam ser, para um usuário, um único sistema.

Com o objetivo de otimizar a comunicação entre computadores que venham a requerir os dados na rede, esse trabalho visa estudar o tempo de comunicação entre máquinas que utilizam sockets para tal finalidade.

1.1 Teoria

O TCP (Transmission Control Protocol) é um protocolo do nível da camada de transporte [4]. Dentre usas principais características temos:

Orientado à conexão Cliente e o servidor trocam pacotes de controle entre si antes de enviarem os pacotes de dados. Isto é chamado de procedimento de estabelecimento de conexão (handshaking).

Transferência garantida Dados trocados são livres de erro, o que é conseguido a partir de mensagens de reconhecimento e retransmissão de pacotes.

Controle de fluxo Assegura que nenhum dos lados da comunicação envie pacotes rápido demais, pois uma aplicação em um lado pode não conseguir processar a informação na velocidade que está recebendo.

controle de congestão Ajuda a prevenir congestionamentos na rede.

Fim-a-fim Conexão é sempre entre o host emissor e o host receptor.

Cada um dos segmentos da camada transporte tem em seu cabeçalho campos denominado número de portas que indicam a qual processo o mesmo deve ser entregue, existindo um número de porta do emissor e o número de porta do receptor.

Possuindo as duas portas, pode-se realizar uma conexão entre elas conhecida por socket. Com o socket é possível diferenciar os canais utilizados por cada processo de aplicação.

Os segmentos TCP, através do protocolo IP, são entregues a sistemas terminais, cada um identificado por seu endereço IP. E o protocolo TCP cuida de repassar os dados à cada processo de aplicação de acordo com porta especificada no cabeçalho do segmento.

Devido a importância do protocolo, este laboratório tem o objetivo de medir o tempo total e de comunicação de uma conexão TCP entre um cliente e um servidor.

2 Servidor de agenda

O sistema implementado, uma agenda distribuída, se baseia numa comunicação cliente-servidor. Nele o servidor possui todas as informações da agenda que estão armazenadas em um banco de dados, assim como as opções de interações com os dados que são apresentadas aos clientes em formas de um menu. O cliente só escolhe alguma opção de interação com os dados de acordo com menu.

2.1 Menu inicial

No menu inicial pode-se:

- Logar
- Criar novo usuário
- Sair

2.1.1 Login

O servidor pede ao usuário o nome de usuário, caso o nome estiver no banco de dados ele pede uma senha que é comparada ao valor do banco de dados, se o usuário não existir é avisado sobre a inexistência, se a senha não conferir é avisado que a senha não confere, caso contrário o usuário consegue logar no sistema, e o servidor recupera sua agenda (cada usuário possui sua agenda).

2.1.2 Novo usuário

O servidor pede um nome de usuário, o servidor verifica se o nome já não existe, se não existir pede a senha e armazena o usuário no sistema, assim como cria uma agenda vazia para o mesmo.

2.2 Menu usuário

Dentre as possibilidades de interações para um usuário logado tem-se:

- Inserção de um compromisso que possui um nome, dia, hora, e minuto.
- Remoção de um compromisso através de seu nome
- Pesquisa de compromisso por dia

- Pesquisa de compromisso por dia e hora
- Ver todos os compromisso de mês de abril

2.2.1 Inserção de compromisso

O usuário deve fornecer o nome do compromisso, o dia, a hora e o minutos em que ele ocorrerá. Caso o compromisso seja possível de ser alocado o servidor avisa com um “OK”, se não for possível também é avisado de tal impossibilidade. Um compromisso é inserido ordenado na agenda se não existir um compromisso com mesmo horário.

2.2.2 Remoção de compromisso

O usuário deve fornecer o nome do compromisso que deve ser removido. Caso o compromisso seja encontrado ele é removido, caso contrário é dito que tal compromisso não existe. Se existirem dois compromissos de mesmo nome, o primeiro é removido. Logo é esperado que compromissos possuam nomes diferentes.

2.2.3 Pesquisas

O servidor faz um requerimento interativo, ou seja, se for selecionado a pesquisa por dia e hora, o servidor pergunta primeiramente o dia e depois a hora. Logo, é uma pesquisa em etapas no qual o servidor interage com nosso usuário.

3 Ambiente de implementação

O sistema de agenda foi implementado e executado nos seguintes sistemas operacionais :

- FC14 - Fedora Laughlin Linux 2.6.35.11

O sistema de agenda foi implementado na linguagem C. Para o armazenamento dos dados, utilizou-se arquivos. Cada usuário possui um arquivo, a sua agenda, no qual armazena-se o nome do compromisso, o dia, a hora e o minuto do mesmo. O sistema lê esse arquivo quando o usuário loga e transfere-o à memória principal, e a cada alteração na agenda o servidor atualiza as informações dos arquivos.

O servidor aceita diversas conexões de clientes, funcionando perfeitamente para interações em diferentes agendas, pois cada cliente possui além de um processo único, que foi criado em um fork, possui um ponteiro para sua agenda. Assim, o servidor consegue alterar todas as agendas independentemente.

O nosso sistema, além disso,apresenta transparência ao usuário. Os tipos de transparência a serem destacados são:

Acesso: Esconde as diferenças nas representações de dados e na invocação de funções ou métodos para facilitar a comunicação entre objetos da rede.

Localização: Esconde o local em que o objeto se encontra.

Concorrência: Esconde como as atividades são coordenadas entre os objetos para obter consistência em um nível mais alto.

Listaremos a seguir algumas funções implementadas de interação:

- Funções de interação com o banco de dados são:

```
/* Encontra usuario que ja esta cadastrado no servidor e verifica
senha*/
int findUser(char nome[], char pwd[]);

/* Insere novo usuario (nome e senha) no banco de dados*/
int newUser(char nome[], char senha[]);

/*Carrega agenda de usuario*/
int loadCal(User *user);

/*Salva agenda*/
int saveCal(User *user);
```

- Funções de interação com a agenda são:

```
/*Cria agenda para usuario*/
User * agenda_init(char nome[]);

/*Apaga agenda da memoria principal do servidor */
void user_destroy(User *u);

/*Insere compromisso na agenda*/
int set_task(int dia,int hora, int min,char task[], User *u);

/*Cria compromisso */
Agenda * task_init(int dia,int hora, int min,char task[]);

/*Retorna compromissos do mes de abril*/
int verMes(int new_fd, User *u);

/*Retorna compromissos do mes de abril em determinado dia*/
int verDia(int new_fd, User *u, int dia);

/*Retorna compromissos do mes de abril em determinado dia e
determinada hora*/
int verHora(int new_fd, User *u, int dia, int hora);

```

```
23      /*Remove compromisso da agenda pelo nome*/
24      int delTask( User *u, char nome[]);
25
26      /*Comapara data de compromissos*/
27      int compData(Agenda *newTasks,Agenda *tasks);
```

- Funções de interação servidor-cliente criadas foram:

```
1
2       /*Envia mensagem ao cliente*/
3       void sendStr(int sockfd, char str[]);
4
5       /*Le mensagem do cliente*/
6       int leOpcao(struct sockaddr_storage their_addr, int sockfd);
7
8       /*Apresenta opcoes de login ou criacao novo usuario*/
9       void menu(int new_fd, struct sockaddr_storage their_addr);
10
11      /*Apresenta opcoes de interacao com agenda*/
12      void menu2(int new_fd, struct sockaddr_storage their_addr, User *user);
13
14      /*Envia mensagem para servidor*/
15      void envia_pct( int sockfd, char s[], int size){
```

4 Tempos de comunicação e total

O round-trip time (RTT) é o tempo que leva-se para um sinal ser enviado mais o tempo que se leva para receber um acknowledgment que o sinal foi recebido. A ferramenta administrativa para as redes de computadores denominada “Ping” [3] é usada para testar se um host é alcançável e para medir o RTT para mensagens enviadas do host remetente para o destinatário. Utilizado o ping nas máquinas nos quais servidor e cliente foram usadas obtivemos os valores apresentados na tabela I.

Valor	Tempo
Max	0.688 ms
Min	0.309 ms
Média	0.513 ms
Desvio	0.086 ms

Tabela I: Tempo do ping

Posteriormente, foi calculado o tempo que o servidor leva para se conectar com o servidor, como é mostrado na tabela II. Os valores foram obtidos de um conjunto de 100 medições.

Valor	Tempo
Max	0.634 ms
Min	0.192 ms
Média	0.517 ms
Desvio	0.081 ms

Tabela II: Tempo de conexão com o servidor

Aplicamos o cálculo de tempo ao programa principal de forma a obtermos o tempo total, tempo de comunicação e os tempos da execução de cada função. Para o tempo total, no cliente pega-se o tempo antes do primeiro send e após o último recv. Para o tempo de comunicação, pega-se o tempo total e subtrai-se o tempo de processamento do servidor, que é depois do primeiro recv e antes do último send.

Para o tempo total das funções obteu-se o tempo de inserir um compromisso, remover o compromisso, ver a agenda do mês, ver a agenda de um dia, ver a agenda de uma hora. Os dados e os testes estão exemplificados nas tabelas III, IV, V, VI e VII.

O resultado obtido para 100 valores foi:

Valor	Tempo
Max	185.351 ms
Min	45.577 ms
Média	55.090 ms
Desvio	0.743 ms

(a) Tempo total

Valor	Tempo
Max	1.483 ms
Min	0.500 ms
Média	0.725 ms
Desvio	0.076 ms

(b) Tempo de comunicação

Tabela III: Conexão e fechamento de conexão com servidor

Valor	Tempo
Max	114.572 ms
Min	32.248 ms
Média	36.782 ms
Desvio	0.069 ms

(a) Tempo total

Valor	Tempo
Max	0.837 ms
Min	0.386 ms
Média	0.705 ms
Desvio	0.001 ms

(b) Tempo de comunicação

Tabela IV: Conexão, login na conta, inserção de compromisso e fechamento de conexão com servidor

Valor	Tempo
Max	214.583 ms
Min	37.724 ms
Média	47.599 ms
Desvio	0.468 ms

(a) Tempo total

Valor	Tempo
Max	0.851 ms
Min	0.515 ms
Média	0.715 ms
Desvio	0.004 ms

(b) Tempo de comunicação

Tabela V: Conexão, login na conta, ver compromissos de determinado dia, hora e minuto e fechamento de conexão com servidor

Note que os tempos das tabelas Va, VIa e VIIa estão, respectivamente, em ordem decrescente de tamanho. Isso se

Valor	Tempo
Max	168.404 ms
Min	32.763 ms
Média	38.612 ms
Desvio	0.412 ms

(a) Tempo total

Valor	Tempo
Max	1.472 ms
Min	0.523 ms
Média	0.727 ms
Desvio	0.002 ms

(b) Tempo de comunicação

Tabela VI: Conexão, login na conta, ver compromissos de determinado dia e hora e fechamento de conexão com servidor

Valor	Tempo
Max	76.481 ms
Min	28.966 ms
Média	32.334 ms
Desvio	0.215 ms

(a) Tempo de total

Valor	Tempo
Max	0.905 ms
Min	0.504 ms
Média	0.716 ms
Desvio	0.001 ms

(b) Tempo de comunicação

Tabela VII: Conexão, login na conta, ver todos os compromissos do mês e fechamento de conexão com servidor

deve ao fato de que a operação 3 (tabela Va) o programa busca pelo dia, hora e minuto do compromisso, fazendo com que o número de comparações feitas seja maior que as operações VIa e VIIa, tornando aquela operação mais lenta. O mesmo pode afirmar entre as operações 4 e 5, correspondentes às tabelas VIa e VIIa, respectivamente.

4.1 Comparação de tecnologias

Como uma das características do TCP é a transferência garantida não foi necessário uma análise de erros na entrega dos pacotes. No entanto, essa análise deve ser feita se utilizássemos o protocolo UDP, através de um código de verificação. Portanto o protocolo TCP possibilita uma diminuição do código.

Para que o nosso sistema distribuído fosse transparente ao usuário diversas manipulações de dados foram realizadas no servidor, assim algumas transparências, como localidade, acesso e concorrência foram possíveis.

Como o TCP é orientado à conexão, é criado um socket para cada cliente se comunicar com o servidor e ele é livre de erros. Utilizando a função fork temos um processo que fica à escuta de novos pedidos de conexão e outro processo trata da conexão com o cliente. Já o UDP não é orientado a conexão, não é confiável e o envio de pacotes pode chegar em diferentes ordens.

Devido as diferentes características do UDP como, por exemplo, o não há tratamento de erros e não ser orientado a conexão, é esperado que ele tenha um tempo inferior ao TCP. Mas, deve-se lembrar que os erros deverão ser verificados pelo programa principal, assim como o ordenamento das mensagens.

O tempo de processamento (manipulação da agenda) teoricamente não deve alterar pela utilização dos diferentes protocolos.

Portanto, é esperado o tempo total será alterado devido ao tempo de comunicação.

5 Conclusão

Segundo nossas medições, os tempos de processamento são significativamente (cerca de duas ordens de grandeza) maiores que os tempos de conexão, ou seja, há uma transparência de localidade para o cliente, já que o tempo de comunicação é tão pequeno que a agenda será aparentemente local para o cliente. Notamos também que todo pacote enviado é recebido, tanto pelo cliente quanto pelo servidor. O que comprova a característica de garantia de transferência do TCP

Referências

[1] Tanenbaum, Andrew S. e Maarten Van Steen Distributed Systems: Principles and Paradigms. Prentice Hall.

[2] Brian "Beej Jorgensen"Hall Beej's Guide to Network Programming - Using Internet Sockets . Disponível em *http://beej.us/guide/bgnet/*, [Último acesso: 07/04/2011].

[3] Mike Muuss Packet Internet Grouper (Groper) . Disponível em *http://linux.die.net/man/8/ping*, [Último acesso: 10/04/2011].

[4] J. Kurose e K. Ross. *Computer Networking: A Top-Down Approach Featuring the Internet*. Pearson Addison Wesley, 3 ed., 2005.