



UNIVERSIDADE FEDERAL DE MINAS GERAIS
DEPARTAMENTO DE CIÊNCIA DA
COMPUTAÇÃO

REDES DE COMPUTADORES
TRABALHO PRÁTICO 3

sistema de mensagens

Aluno:
Pedro Araujo Pires

Professor:
Luis Felipe Menezes Vieira

30 de junho de 2012

Sumário

1	Introdução	2
1.1	Modelo cliente-servidor	2
1.2	Protocolo TCP	2
1.3	Sistema de mensagens	3
2	Implementação	3
2.1	Mensagens	3
2.2	Servidor	4
3	Conclusões	4

1 Introdução

Já fazem muitos anos que não conseguimos pensar em um computador como uma máquina isolada do mundo. As redes permitem a comunicação entre computadores, e por consequência entre usuários. A Internet não nos deixa dúvidas.

1.1 Modelo cliente-servidor

Grande parte da comunicação entre processos é feita utilizando o modelo cliente-servidor. Este modelo se baseia na ideia de que um processo (o cliente) se conecta a outro (o servidor) para pedir ou enviar informações. Uma boa analogia seria uma pessoa fazendo uma ligação telefônica para outra. A pessoa que faz a ligação precisa de saber o número do telefone da outra, mas quem recebe não sabe quem está ligando. Uma vez que a chamada é completada, ambas as pessoas podem falar e/ou escutar. A figura 1.1 ilustra este modelo.

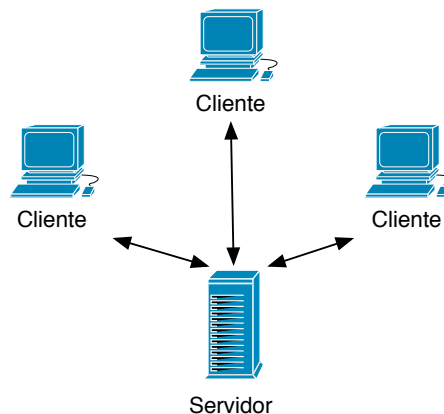


Figura 1: Modelo cliente-servidor

1.2 Protocolo TCP

Para trocar informações entre computadores, além de estarem conectados, eles precisam entender o que representam os dados enviados através da rede. Este é o papel dos protocolos de rede. A Internet foi projetada através de camadas de protocolos. Deste modo os desenvolvedores teriam liberdade para criar e mudar as formas de comunicação sem precisar remodelar toda a estrutura da Internet. O protocolo TCP - Transmission Control Protocol - fica na camada de transporte, como visto na figura 1.2. Isto significa que ele é responsável por passar uma mensagem vinda da rede para uma aplicação um nível acima. O TCP utiliza a forma de fluxo de bytes para enviar mensagens e garante a confiabilidade do canal. Desta forma a aplicação não precisa de se responsabilizar pelo tratamento de possíveis erros na transmissão de pacotes entre o cliente e o servidor.

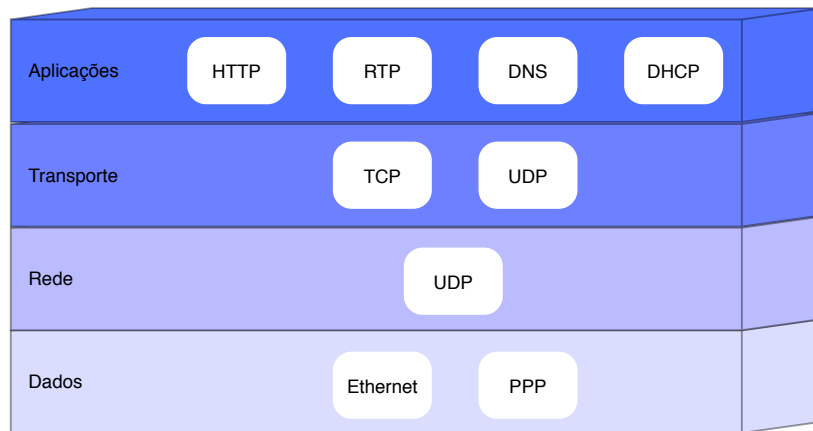


Figura 2: Camadas de protocolos

1.3 Sistema de mensagens

O objetivo deste trabalho prático foi implementar um sistema de troca de mensagens utilizando o modelo cliente-servidor e protocolo TCP. Para isso 3 aplicações foram desenvolvidas:

- Servidor: Responsável por receber as conexões dos clientes, e fazer a troca de mensagens entre os clientes.
- Cliente de exibição: Exibe as mensagens destinadas a este cliente.
- Cliente de envio: Envia mensagens para outros clientes.

2 Implementação

2.1 Mensagens

As mensagens enviadas pela aplicação são enviadas na forma de sequência de bytes. Os 8 primeiros bytes são usados no cabeçalho. Os bytes subsequentes contêm o corpo da mensagem, que pode ter até 141 caracteres, sendo o último reservado para o caractere de término de string. O formato do cabeçalho é descrito a seguir:

- **Tipo:**
 - OI (0): Mensagem utilizada para um cliente se identificar para o servidor, e abrir uma conexão entre os dois.
 - TCHAU (1): Mensagem especial para o cliente dizer ao servidor que vai fechar a conexão.
 - MSG (2): Mensagem contendo um texto que pode ser enviado para um cliente específico ou para todos os clientes conectados ao servidor.

- ERRO (3): Mensagem retornada pelo servidor quando um cliente tenta se conectar com um ID em uso por outro cliente.

- **Origem:** ID do cliente que enviou a mensagem.
- **Destino:** ID do cliente de destino.
- **Tamanho:** Tamanho do corpo da mensagem.



Figura 3: Formato da mensagem

2.2 Servidor

O servidor é o responsável receber as mensagens dos clientes de envio e enviá-las para seu destinatário. Caso chegue uma mensagem com o identificador do origem errado, o servidor descarta a mensagem.

Uma vez por minuto o servidor envia uma mensagem para todos os clientes de exibição informando a quantidade de clientes conectados e o tempo desde que o servidor foi iniciado. Para implementar o envio periódico de mensagens foi utilizada uma thread que é iniciada junto com servidor, e é responsável por receber o sinal de alarme e enviar a mensagem com as informações.

3 Conclusões

O trabalho foi de grande importância para conhecer a complexidade na implementação de servidores que suportam conexões de vários clientes simultaneamente. Também foi possível estudar e implementar o temporizador utilizando sinais e funções de manipulação de tempo da biblioteca padrão do Unix. A teoria envolvida neste trabalho é bem simples, entretanto a implementação possui vários detalhes de funcionamento que a torna complexa de se codificar.

Referências

- [1] http://www.linuxhowtos.org/C_C++/socket.htm
- [2] <http://www.slideshare.net/jignesh/socket-programming-tutorial>
- [3] <http://beej.us/guide/bgnet/output/html/multipage/clientserver.html>