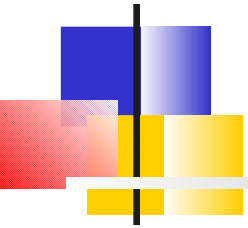
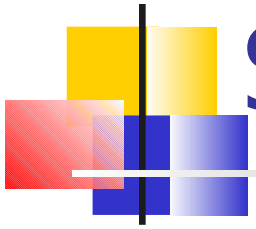


Aula de Socket

Rafael De Tommaso do Valle

20 de agosto de 2009

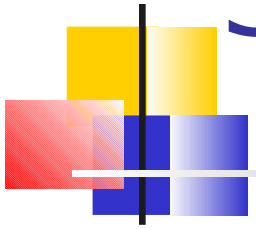


Socket

- O que é um socket?
  - É uma interface com qual processos em diferentes hosts se comunicam através da rede;
  - Também chamado de interface de programação da aplicação (API);
  - Um socket é a interface entre a camada de aplicação e a de transporte da rede.

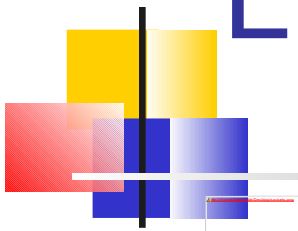
- Internet sockets (nosso interesse):
  - UDP – não orientado à conexão (Datagram Sockets)
  - TCP – orientado à conexão (Stream Sockets)

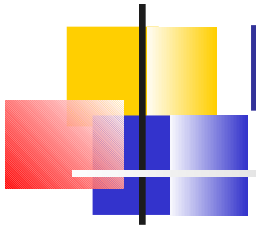
Sockets em C



- Para programação de sockets será utilizada a linguagem C;
- Possui um conjunto de funções para a criação de sockets;
- Os sockets deverão ser executados em Sistemas Unix.

Estrutura





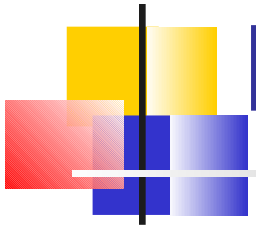
Endereçamento

Estrutura de endereçamento

```
struct sockaddr_in{
        short int sin_family;
        unsigned short int sin_port;
        struct in_addr sin_addr;
        unsigned char sin_zero[8];
                }
```

O **primeiro** item da estrutura define o tipo de família do protocolo a ser usado (AF_INET).

O **segundo** indica o número da porta TCP ou UDP usada na comunicação entre os processos. Para se atribuir o valor a este item, é necessário usar uma função que transforma a representação de dados do host na representação de dados da rede.



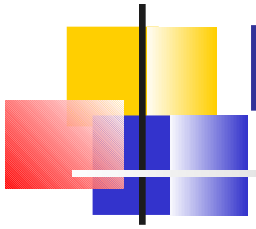
Endereçamento

Mas como fazer isso?

sin.sin_port = htons (SERVER_PORT);
onde SERVER_PORT é o número da porta a ser utilizada.

O **terceiro** item é o endereço IP do host de destino. No caso do servidor, utiliza-se a constante INADDR_ANY. Nos clientes usaremos a função **inet_addr**() ou **gethostbyname**().

O **quarto** item sin_zero existe para zerar a parte da estrutura que não foi usada, já que é alocado espaço para o maior tamanho de endereço possível.

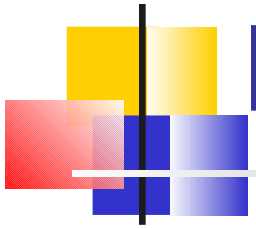


Funções

Criação do socket – Função socket()

int socket(int family, int type, int protocol);

- **int family:** indica a família de protocolos que será utilizada (PF_INET).
- **int type:** define o tipo de socket a ser criado (para UDP, SOCK_DGRAM e para TCP, SOCK_STREAM).
- **int protocol:** identifica o protocolo específico a ser usado. Neste caso será nulo, já que os dois primeiros argumentos já identificam exclusivamente o protocolo.
- Se o socket é criado, retorna o descritor de arquivos para este socket, caso contrário retorna um valor negativo.



Funções

Associação do socket a uma porta – Função bind()

*int bind(int socket, struct sockaddr *address, int addr_len);*

- **int socket:** é o socket criado pela função socket().
- **struct sockaddr *address:** é a estrutura de endereçamento que contém as informações necessárias para o estabelecimento da associação.
- **int addr_len:** é o tamanho dessa estrutura, pois, dependendo da família e do protocolo utilizados, ele varia.
- Retorna um valor negativo em caso de insucesso.

Funções TCP

- **Fila de Conexões Pendentes – Função listen()**

  Instrui o socket a esperar pela conexão de um cliente.

  int listen(int socket, int backlog);

- **Estabelecimento da Conexão – Função accept()**

  Aceita conexão com o cliente.

  *int accept(int socket, struct sockaddr *address, int *addr_len);*

- **Conexão – Função connect()**

  Usado pelo cliente para estabelecer conexão com o servidor

  *int connect(int socket, struct sockaddr *address, int addr_len);*

Função de envio UDP

Envio de mensagens usando UDP – Função sendto()

*ssize_t = sendto(int socket, char *message, int msg_len, int flags, struct sockaddr *address, int addr_len);*

- **char *message:** é o endereço da variável onde se encontra a mensagem que se deseja transmitir pelo socket.
- **int msg_len:** tamanho dessa mensagem.
- **int flags:** é um conjunto de flags que controlam certos detalhes da operação mas que podem receber um valor nulo.
- ***struct sockaddr *address:*** estrutura de endereçamento de destino.
- ***int addr_len:*** tamanho da estrutura de endereçamento (**sizeof** address).
- Retorna o número de bytes enviados ou –1, em caso de erro.

Função de recepção UDP

Recepção de mensagens usando UDP – Função recvfrom()

*ssize_t = recvfrom(int socket, char *buffer, int buffer_len, int flags, struct sockaddr *address, int *addr_len);*

- Argumentos similares a função sendto();
- A função retorna, além do datagrama recebido (no segundo argumento), a estrutura de endereçamento da origem de forma que o destino possa enviar-lhe uma resposta (quinto argumento) e o número de bytes da área de dados recebidos ou –1 em caso de erro.

Funções

Fechando o socket – Função close()

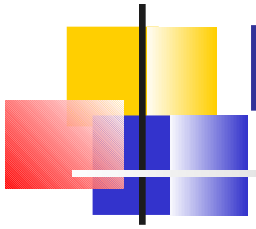
int close(int socket);

A função retorna um valor nulo em caso de sucesso.

Bibliotecas

Para usar estas funções, devem ser incluídas as seguintes bibliotecas:

```
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <netdb.h>
```

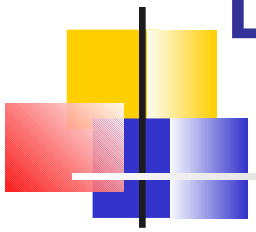


Referências

- **“Computer Networks – A System Approach”** – Larry Peterson & Bruce Davie – Morgan Kauffman – 2ª edição (exemplo de um programa cliente e de um programa servidor usando TCP – Capítulo 1 – Pasta 635 na Xerox da Ângela)

- **“Unix Network Programming”** – Richard Stevens – Prentice Hall – 2ª edição – Volume 1 (referências sobre programação de interfaces socket)
- “**Beej's Guide to Network Programming,** Using Internet Sockets” - Brian "Beej Jorgensen" Hall - http://beej.us/guide/bgnet/output/html/multipage/index.html

Exemplo



- Socket UDP:
  - Dois arquivos:
    - Servidor: que espera por mensagens na porta 5000;
    - Cliente: que envia mensagens a esse servidor.