

Publicidade



ASSINE 0800 703 3000

BATE-PAPO

E-MAIL

SAC

Voip

E-Mail Grátis

Shopping

ÍNDICE PRINCIPAL

PROCURAR:

no site

OK

Terça, 24/02/2009

- » Introdução
- » Programação
- » Administração
- » Hardware
- » Aplicativos
- » Jogos
- » Segurança
- » Editorial
- » Entrevistas

ARTIGOS

- » Fórum
- » Links
- » Notícias
- » Pegue o Linux
- » Documentação

OLINUX SERVIÇOS COMUNIDADE

- » Programas
- » Dúvidas
- » Oportunidades
- » Sobre
- » Contato
- » Publicidade

Powered By:
DEBIAN
GNU/LINUX

English Version

Linux Solutions

Shopping
OLinux

Programação

Tutorial de Sockets - Parte II

Por: [Frederico Perim](#)

► Chamadas de Sistema

Agora iremos abordar as chamadas de sistema que permitem acessar a [funcionalidade](#) de rede em um ambiente Linux. Quando você chama uma dessas funções, o kernel assume o controle e realiza todo trabalho pra você [automaticamente](#).

O problema que a maioria das pessoas tem encontrado é na ordem em que chamar essas funções. Neste aspecto as páginas man são inúteis, como você já deve ter percebido. Para ajudar nessa árdua situação, tentarei abordar as chamadas de sistema nas próximas linhas aproximadamente na mesma ordem em que você precisará chamá-las em seus programas.

► socket() - Acesso ao Descritor de Arquivo!

Imagino que não posso mais adiar. Vou ter que falar sobre a chamada socket().

```
int socket (int domínio, int tipo, int protocolo);
```

Mas o que são estes argumentos? Primeiro, domínio deve ser setado como "AF_INET", assim como struct sockaddr_in (acima). A seguir, o argumento tipo diz ao kernel qual tipo de socket, ou seja: SOCK_STREAM ou SOCK_DGRAM. Finalmente, ajuste protocolo como "0" para que socket() escolha o protocolo correto baseado no argumento tipo. (Nota: Existem vários domínios, Existem vários tipos. Consulte o manual socket() de seu 1. Também existe uma forma melhor de conseguir o protocolo. Consulte getprotobyname()).

Socket simplesmente retorna o descritor que você vai usar em chamadas de sistema mais tarde, ou -1 em caso de erro. A variável global errno é setada com o valor de erro.

Legal, agora continue lendo e faça mais algumas chamadas de sistema para isto fazer sentido.

► Bind() - Em que porta estou?

Uma vez que tenha um socket, você deve associá-lo a uma porta em sua máquina. (Isso é mais comum quando seu programa ouvir (listen()) conexões remotas em uma porta específica.

O número da porta é usado pelo kernel para ajustar um pacote que estiver recebendo a um determinado processo. Se você for somente usar a função connect(), isto pode não ser necessário.

Aqui está uma pequena sinopse da chamada de sistema bind():

```
int bind(int sockfd, struct sockaddr *meu_end, int  
addrlen);
```

sockfd é o descritor de socket retornado por socket(). meu_end é um ponteiro a um struct sockaddr que contém informações de seu endereço (porta e endereço IP). addrlen pode ser setado para sizeof(struct sockaddr).

ENQUETE

Com qual frequência você
acessa o site Olinux?

- ☐ Todos os dias
- ☐ Uma vez por semana
- ☐ Cinco vezes aos mês
- ☐ Poucas vezes ao mês
- ☐ Outra

VOTAR

NEWSLETTER

Inscriva-se e receba as últimas
notícias, programas, artigos,
novidades e tudo do mundo
Linux que aconteceu na semana.

Digite seu email:

OK

Jaguar

Vários modelos.
Entre e confira.

Brinquedos

das Meninas Super
Poderosas. Clique!

Filmadora

Multilaser CR-518
Digital.
Compare!

Esteira

Entre em forma
antes do verão.

COMPARE PREÇOS

Buscar

Opa! Isso é um pouco complicado para absorver de uma vez, então vamos a um exemplo:

```
main()
{
    int sockfd;
    struct sockaddr_in meu_end;

    sockfd = socket(AF_INET, SOCK_STREAM, 0); // realizar
    checagem de erro

    meu_end.sin_family = AF_INET;           // host byte order
    meu_end.sin_port = htons(3490);         // converte em short,
    network byte order
    meu_end.sin_addr.s_addr = inet_addr("10.12.110.57");
    memset(&(meu_end.sin_zero), '\0', 8); // zerar o resto
    da estrutura

    // não se esqueça de realizar checagem de erro em
    bind():

    bind(sockfd, (struct sockaddr *)&meu_end,
    sizeof(struct sockaddr));

}
```

Note que meu_end.sin_port está em "Network Byte Order", assim como meu_end.sin_addr.s_addr. Outra coisa a ser esclarecida é que os [arquivos](#) de cabeçalho(header) podem ser diferentes dependendo de seu sistema. Para ter certeza acesse o man de seu sistema.

[Próximo»](#)

- ▶ Chamadas de Sistema
- ▶ socket() - Acesso ao Descritor de Arquivo!
- ▶ Bind() - Em que porta estou?
- ▶ Bind() - Em que porta estou? (continuação)
- ▶ connect() - Já era hora!



Enviar para um
amigo



Imprimir



Índice de
artigos

Publicidade / Sobre OLinux / Entre em Contato / Privacidade

Copyright (c) 2000-2007, OLinux - O Portal de Linux do Brasil.

Desenvolvido por: [Linux Solutions](#)

Todos os Direitos Reservados.