


Publicidade



UOL | ASSINE 0800 703 3000 | BATE-PAPO | E-MAIL | SAC | Voip | E-Mail Grátis | Shopping | ÍNDICE PRINCIPAL

PROCURAR: no site

Terça, 24/02/2009

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

ARTIGOS

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

COMUNIDADE

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

SERVIÇOS

Powered By:
DEBIAN
GNU/LINUX

English Version

Linux Solutions
Shopping
OLinux



Programação

Tutorial de sockets - Parte VII

Por: [Frederico Perim](#)

► Lidando com send() parciais

Lembra da seção sobre send() quando eu disse que este pode não enviar todos os bytes que você pediu? Ou seja, você quer enviar 512 bytes, mas retorna 412. O que aconteceu com os outros 100 bytes?

Bem, eles ainda estão no [buffer](#) esperando serem enviados. Devido a circunstâncias além do seu controle, o kernel decidiu não enviar os [dados](#) de uma vez, e agora meu amigo, cabe a você enviá-los.

Você poderia escrever uma função com esta para fazer isso:

```
int enviado (int s , char *buff, int *len)
{
    int total = 0 ; // quantos bytes enviamos?
    int bytes_resta = *len; // quantos ainda restam para enviar
    int n;

    while ( total < *len) {

        n = send (s, buf+total, bytes_resta, 0);
        if ( n == -1 ) { break; }
        total += n;
        bytes_resta -= n;
    }

    *len = total; // retorna o número que realmente foi
    enviado

    return n==-1?-1: 0; // retorna -1 em caso de falha, 0 caso
    sucesso
}
```

Nesse exemplo, s é o [socket](#) para onde você quer enviar os dados, buf é o buffer contendo os dados, e len é um ponteiro para um int contendo o número de bytes no buffer.

A função retorna -1 caso ocorra erro. Além disso, o número de bytes realmente enviados é retornado em len. Isso será o mesmo número de bytes que você pediu para enviar, contanto que não ocorra erro. enviado() vai fazer o possível para enviar os dados, mas caso ocorra erro, ela retorna para você imediatamente.

Para completar, aqui está um exemplo de chamada para esta função:

```
char buf[10] = "Hrerer!";
int len;

len = strlen (buf);

if (enviado ( s, buf, &len) == -1) {
```

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

NEWSLETTER

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

Digite seu email:

Relógio

de Pulso em até
12x.

Brinquedos

das Meninas Super
Poderosas. Clique!

Vinhos

Diversas marcas a
partir de R\$ 9,90!
Aproveite!

Esteira

Entre em forma
antes do verão.

COMPARE PREÇOS

Buscar

```
perror ("enviado");  
printf ("Nós somente enviamos %d bytes porque houve  
erro!!\n", len);  
}
```

O que acontece no lado do receptor quando parte do pacote chega? Se os pacotes são de tamanhos variáveis, como o receptor sabe quando um pacote termina e outro começa? Sim, cenários reais são chatos assim. Você provavelmente vai ter que empacotar (lembra-se da seção sobre encapsulamento no começo do documento?) Continue lendo para mais detalhes.

[Próximo»](#)

▶ **Lidando com send() parciais**

▶ **Filho do Empacotamento de Dados**



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.