


© 2002-2015 Volnys Bernal 1

Cliente UDP

Volnys Borges Bernal
volnys@lsi.usp.br

Departamento de Sistemas Eletrônicos
Escola Politécnica da USP




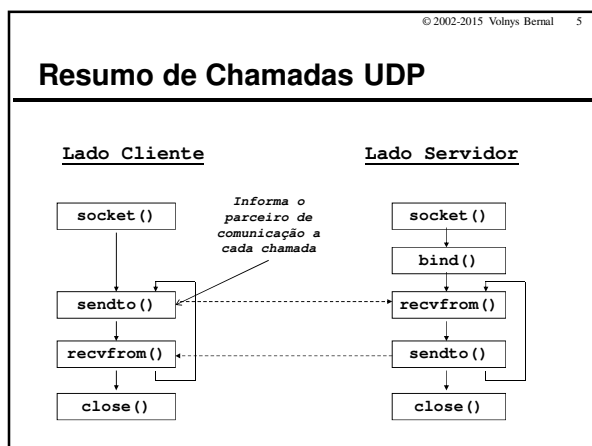
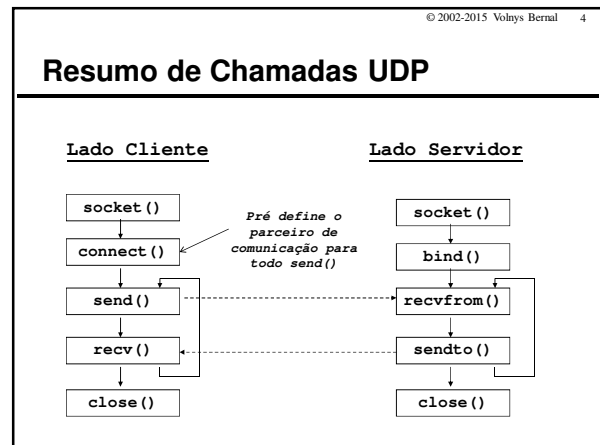
© 2002-2015 Volnys Bernal 2

Agenda

- ❑ Resumo das chamadas sockets para UDP
- ❑ Uso das chamadas sockets para UDP
 - ❖ Chamada socket()
 - ❖ Chamada connect()
 - ❖ Chamada send()
 - ❖ Chamada recv()
 - ❖ Chamada close()


© 2002-2015 Volnys Bernal 3

Resumo das chamadas UDP

© 2002-2015 Volnys Bernal 6

Chamada socket()



© 2002-2015 Volnys Bernal 7

Chamada socket()

- ❑ **Objetivo**
 - ❖ Criar um novo socket (plug de comunicação). Aloca estruturas de dados no sistema operacional para suportar a comunicação.
- ❑ **Resultado**
 - ❖ Retorna o descritor de arquivo (número inteiro).
- ❑ **Sintaxe**

```
sd = socket (int domain, int type, int protocol)
```
- ❑ **Observação:**
 - ❖ Quando um socket é criado, não possui nenhuma informação sobre o parsocket (endereços IPs e portas dos parceiros).

© 2002-2015 Volnys Bernal 8

Chamada socket()

- ❑ **Sintaxe geral**

```
#include <sys/socket.h>
int socket(int domain, int type, int protocol)
```

Socket descriptor

Pilha de protocolos:
 - PF_LOCAL (file)
 - PF_INET (IPv4)
 - PF_INET6 (IPv6)
 - PF_X25 (X25)

Tipo da comunicação:
 - SOCK_STREAM (TCP)
 - SOCK_DGRAM (UDP)
 - SOCK_RAW (IP)

Para PF_INET use 0
 - Se SOCK_DGRAM - UDP
 - Se SOCK_STREAM - TCP

© 2002-2015 Volnys Bernal 9

Chamada socket()

- ❑ **Tipo de serviço**
 - ❖ SOCK_STREAM
 - Para ser utilizado com o protocolo TCP
 - Canal de comunicação full duplex
 - Fluxo de bytes sem delimitação
 - Chamadas para transmissão e recepção de dados:
 - read(), write() ou send(), recv()
 - ❖ SOCK_DGRAM
 - Para ser utilizado com o protocolo UDP
 - Datagrama (mensagens)
 - Chamadas para transmissão e recepção de dados:
 - send(), sendfrom(), recv() ou recvfrom()
 - ❖ SOCK_RAW
 - Permite acesso a protocolos de mais baixo nível
 - Datagrama (mensagens)
 - Chamadas para transmissão e recepção de dados:
 - send(), recv()

© 2002-2015 Volnys Bernal 10

Chamada socket()

- ❑ **Para criar um socket TCP**

```
#include <sys/socket.h>
sd = socket(AF_INET, SOCK_STREAM, 0);
```
- ❑ **Para criar um socket UDP**

```
#include <sys/socket.h>
sd = socket(AF_INET, SOCK_DGRAM, 0);
```

© 2002-2015 Volnys Bernal 11

Chamada socket()


- ❑ **Exemplo de criação de socket UDP**

```
#include <sys/socket.h>

int sd; // socket descriptor
...
sd = socket(PF_INET, SOCK_DGRAM, 0);
if (sd == -1)
{
    perror("Erro na chamada socket");
    exit(1);
}
...
```

© 2002-2015 Volnys Bernal 12

Chamada Connect()



© 2002-2015 Volnys Bernal 13

Chamada connect()

- ❑ **Objetivo**
 - ✦ Estabelecer uma sessão de comunicação TCP, UDP ou IP
- ❑ **Detalhamento**
 - ✦ Deve ser utilizado somente no lado cliente
 - ✦ UDP:
 - Informa ao sistema operacional o socket address (IP+porta) do parceiro de comunicação
 - Não são enviados datagramas
 - ✦ TCP:
 - Informa ao sistema operacional o socket address (IP+porta) do parceiro de comunicação
 - Estabelece a conexão TCP (3 way handshake)

© 2002-2015 Volnys Bernal 14

Chamada connect()

- ❑ **Sintaxe**

```
#include <netdb.h>
int connect(int sd,
            struct sockaddr *serversockaddr,
            int size)
```

Diagram illustrating the arguments of the `connect()` function:

- `sd`: Socket descriptor
- `serversockaddr`: Socket address (IP + porta) do parceiro (servidor)
- `size`: Tamanho da estrutura de endereço (sockaddr_in)

© 2002-2015 Volnys Bernal 15

Chamada connect()

```
#include <netdb.h>

int          status;           //estado da chamada
struct sockaddr_in serveraddr; //endereço do servidor
...

// define endereço destino
serveraddr.sin_family = AF_INET;
serveraddr.sin_port   = htons(serverport);
status = inet_pton(AF_INET, stringIP, &serveraddr.sin_addr);
if (status <= 0)
    perror("Erro na conversão do endereço IP");

// ativa connect
status = connect( sd,
                 (struct sockaddr *)&serveraddr,
                 sizeof(serveraddr) );
if (status != 0)
    perror("Erro na chamada connect");
```

© 2002-2015 Volnys Bernal 16

Chamada connect


- ❑ **Exemplo de iniciação da estrutura sockaddr_in**

Diagram illustrating the initialization of the `sockaddr_in` structure:

- `AF_INET` is assigned to `sin_family`.
- `80` is converted to network byte order using `htons()` and assigned to `sin_port`.
- `"10.0.0.1"` is converted to binary form using `inet_pton()` and assigned to `sin_addr`.
- `sin_zero` is shown as a byte array.

© 2002-2015 Volnys Bernal 17

Chamada send()



© 2002-2015 Volnys Bernal 18

Chamada send()

- ❑ Função para transmissão de dados
- ❑ Pode ser utilizada por clientes e servidores

```
int send(int sd, void *txbuffer, int msgsize, int flags)
```

Diagram illustrating the arguments of the `send()` function:

- `sd`: Socket Descriptor
- `txbuffer`: Ponteiro para buffer da mensagem (end. do buffer da mensagem)
- `msgsize`: Tamanho da mensagem
- `flags`: Opções

© 2002-2015 Volnys Bernal 19

Chamada send()


❑ Exemplo:

```
char txbuffer[80];

...
status = send (sd, txbuffer, strlen(txbuffer)+1, 0)
if (status < 0)
    perror("Erro na chamada send");
...
```

© 2002-2015 Volnys Bernal 20

Chamada recv()



© 2002-2015 Volnys Bernal 21

Chamada recv()

❑ Recebimento de datagramas
❑ Pode ser utilizada por clientes e servidores

```
int recv(int sd, void * rxbuffer, int rxbuffersize, int flags)
```

Diagram illustrating the parameters of the `recv` function:

- `sd`: Socket Descriptor
- `rxbuffer`: Ponteiro para o buffer (end. do buffer de recepção)
- `rxbuffersize`: Tamanho do buffer
- `flags`: Opções

© 2002-2015 Volnys Bernal 22

Chamada recv()

❑ Exemplo:

```
char rxbuffer[80];

...
status = recv(sd, rxbuffer, sizeof(rxbuffer), 0)
if (status < 0)
    perror("Erro na chamada recv");
printf("MSG recebida: %s\n", rxbuffer);
...
```

© 2002-2015 Volnys Bernal 23

Chamada recv()

❑ Bloqueante


- ❖ Se não existirem mensagens na fila de recepção o processo fica aguardando sua chegada
- ❖ Exceção: quando o socket for criado como não bloqueante (ver `fcntl(2)`).

❑ Retorno

- ❖ Se a chamada tiver sucesso, o valor retornado é o tamanho do datagrama

© 2002-2015 Volnys Bernal 24

Chamada close()



© 2002-2015 Volnys Bernal 25

Chamada close()

- ❑ **Objetivo**
 - ❖ Fechar o descritor de arquivos (neste caso, fecha o socket).
 - ❖ Se ainda existirem dados para serem transmitidos pelo socket, aguarda por alguns segundos a finalização desta transmissão.
- ❑ **Resultado**
 - ❖ Fecha o descritor do arquivo.
- ❑ **Sintaxe**

```
int close (int sd)
```

© 2002-2015 Volnys Bernal 26

Chamada close()

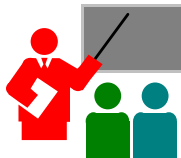
- ❑ **Exemplo:**

```
int sd; // socket descriptor
...

status = close(sd);
if (status == -1)
    perror("Erro na chamada close");
...
```

© 2002-2015 Volnys Bernal 27

Exercício



© 2002-2015 Volnys Bernal 28

Exercício

- (1) Identifique a porta utilizada no serviço “echo”.
- (2) Implemente um cliente para o serviço “echo” utilizando o protocolo UDP.
 - ❖ O serviço echo responde exatamente com a sequência ASCII recebida.


© 2002-2015 Volnys Bernal 29

Exercício

- (3) Identifique a porta utilizada no serviço “daytime”.
- (4) Implemente um cliente para o serviço daytime utilizando o protocolo UDP.
 - ❖ O serviço daytime UDP responde com a data e hora do servidor no instante de recebimento do datagrama UDP.

© 2002-2015 Volnys Bernal 30

Referências Bibliográficas



Referências Bibliográficas

- **COMMER, DOUGLAS; STEVENS, DAVID**
 - ❖ Internetworking with TCP/IP: volume 3: client-server programming and applications
 - ❖ Prentice Hall
 - ❖ 1993