


© 2004-2015 Volnys Bernal 1

Servidor TCP


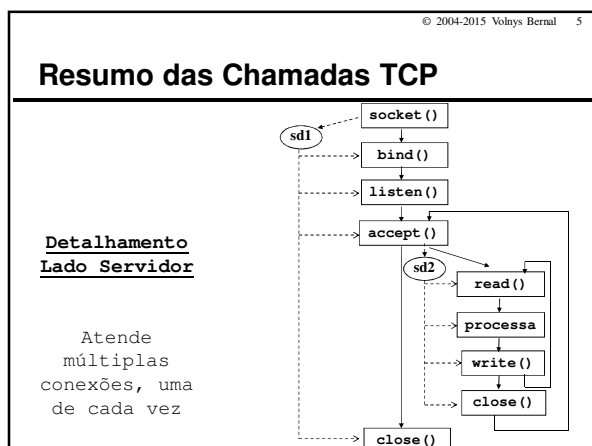
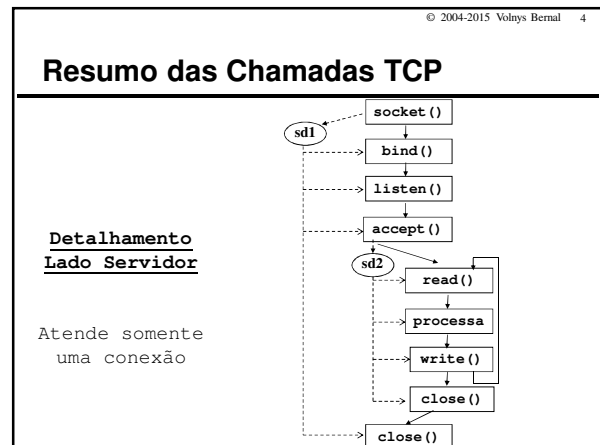
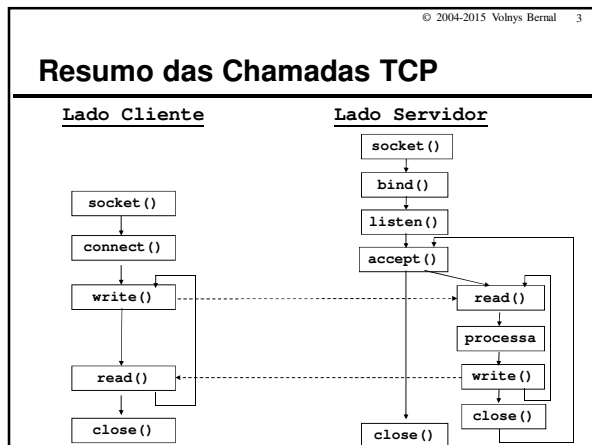
Volnys Borges Bernal
volnys@lsi.usp.br

Departamento de Sistemas Eletrônicos
Escola Politécnica da USP




© 2004-2015 Volnys Bernal 2

Resumo das Chamadas TCP

© 2004-2015 Volnys Bernal 6

Chamada socket()



© 2004-2015 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).

© 2004-2015 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

© 2004-2015 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()

© 2004-2015 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);
```

© 2004-2015 Volnys Bernal 11

Chamada socket()


- ❑ **Exemplo de criação de socket TCP**

```
#include <sys/socket.h>
int sd;

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

© 2004-2015 Volnys Bernal 12

Chamada bind()



© 2004-2015 Volnys Bernal 13

Chamada bind()

- ❑ **Objetivo**
 - ❖ Definir o *socket address* local (end. IP e porta TCP)
 - ❖ Deve ser utilizado somente no lado servidor
- ❑ **Valor retornado pela função**
 - ❖ 0: sucesso
 - ❖ -1: erro

© 2004-2015 Volnys Bernal 14

Chamada bind()

- ❑ **Sintaxe**

```
#include <netdb.h>

int bind(
    int sd,
    struct sockaddr *myaddr,
    socklen_t addrlen)

```

Socket address local: Endereço IP da interface local e e porta TCP pela qual o servidor TCP irá aguardar conexões.

Resultado da chamada: sucesso ou erro

Descritor do socket

Tamanho da estrutura de endereço (sockaddr)

© 2004-2015 Volnys Bernal 15

Chamada bind()

- ❑ **Socket address local**
 - ❖ Pode ser o endereço IP de:
 - Uma interface específica
 - Todas interfaces locais (INADDR_ANY)

© 2004-2015 Volnys Bernal 16

Chamada bind()

```
#include <netdb.h>


struct sockaddr_in mylocal_addr;
...
mylocal_addr.sin_family = AF_INET;
mylocal_addr.sin_addr.s_addr = INADDR_ANY;
mylocal_addr.sin_port = htons(myport);

status = bind(socketdescriptor,
              (struct sockaddr *) &mylocal_addr,
              sizeof(struct sockaddr_in));

if (status == -1)
    perror("Erro na chamada bind");
```

© 2004-2015 Volnys Bernal 17

Chamada listen()



© 2004-2015 Volnys Bernal 18

Chamada listen()

- ❑ **Objetivo**
 - ❖ Abrir a porta na qual o servidor irá aguardar conexões TCP
 - ❖ As conexões são aceitas através da ativação de accept()
- ❑ **Valor retornado pela função**
 - ❖ 0: sucesso
 - ❖ -1: erro

© 2004-2015 Volnys Bernal 19

Chamada listen()

□ Sintaxe

Resultado da função:

0: sucesso

-1: erro

Descritor do socket

```
int listen(int sd,
          int queuelenght)
```

Qde máxima de conexões pendentes sem aceite

© 2004-2015 Volnys Bernal 20


Chamada listen()

□ Exemplo:

```
#define QLEN 10
...
status = listen(sd, QLEN);
if (status != 0)
{
    perror("Erro na chamada listen()");
    exit(1);
}
```

© 2004-2015 Volnys Bernal 21

Chamada accept()



© 2004-2015 Volnys Bernal 22

Chamada accept()

□ Objetivo

- ❖ Aceitar uma nova conexão TCP.
- ❖ Accept() extrai a primeira conexão da fila e gera um novo *socket descriptor* para esta conexão.
- ❖ O socket original não é afetado por esta chamada.

□ Valor retornado pela função

- ❖ Valor não negativo: sucesso, sendo este o valor do novo socket descriptor
- ❖ -1: erro

© 2004-2015 Volnys Bernal 23

Chamada accept()

□ Sintaxe

Valor retornado:

Positivo: sucesso, correspondendo ao valor do novo socket

-1: erro

Descritor do socket

```
int accept (int sd,
            struct sockaddr *addr,
            socklen_t *addrlen)
```

Tamanho da estrutura sockaddr (precisa ser uma variável !)

Ponteiro para uma estrutura sockaddr que conterá o endereço (socket address) do parceiro de comunicação

© 2004-2015 Volnys Bernal 24

Chamada accept()

□ Exemplo

```
int newsd;
int size;
struct sockaddr_in clientaddr;


....
size = sizeof(clientaddr);
newsd = accept( sd,
                (struct sockaddr *) &clientaddr,
                (socklen_t *) &size);

if (newsd < 0)
{
    perror("Erro na chamada accept()");
    exit(1);
}

....
```

© 2004-2015 Volnys Bernal 25

Chamada read()



© 2004-2015 Volnys Bernal 26

Chamada read()

- ❑ **Objetivo**
 - ❖ Recepção/leitura de dados de um descritor
 - Descritor: descritor sockets, descritor de arquivo, ...
 - ❖ Pode ser utilizada por cliente ou servidor
- ❑ **Valor retornado pela função**
 - ❖ >0: quantidade de bytes lidos
 - ❖ 0: end of file
 - ❖ -1: erro

© 2004-2015 Volnys Bernal 27

Chamada read()

❑ **Sintaxe:**

```
#include <unistd.h>

int read(int sd, void *buf, int buffersize)
```

Socket
Descriptor

Tamanho do
buffer

Ponteiro para o buffer
(end. do buffer de recepção)

© 2004-2015 Volnys Bernal 28

Chamada read()


❑ **Exemplo:**

```
char rxbuffer[80];

...
status = read(newsd, rxbuffer, sizeof(rxbuffer))
if (status == -1)
    perror("Erro na chamada read");
printf("MSG recebida: %s\n", rxbuffer);
...
```

© 2004-2015 Volnys Bernal 29

Chamada write()



© 2004-2015 Volnys Bernal 30

Chamada write()

- ❑ **Objetivo**
 - ❖ Transmissão/escrita de dados em um descritor
 - Descritor: descritor sockets, descritor de arquivo, ...
 - ❖ Pode ser utilizada por cliente ou servidor
- ❑ **Valor retornado pela função**
 - ❖ Positivo: quantidade de bytes escritos
 - ❖ -1: erro

© 2004-2015 Volnys Bernal 31

Chamada write()

❑ Sintaxe

```
#include <unistd.h>

int write(int sd, void *buf, int count)
```

© 2004-2015 Volnys Bernal 32

Chamada write()

❑ Exemplo:

```
#include <unistd.h>
char txbuffer[80];

...
status = write(newsd, txbuffer, strlen(txbuffer)+1)
if (status == -1)
    perror("Erro na chamada write");
...
```

© 2004-2015 Volnys Bernal 33

Chamada close()

© 2004-2015 Volnys Bernal 34

Chamada close()

- ❑ Permite fechar o socket (assim como é feito com arquivo)
- ❑ Existem dois sockets abertos. É importante fechar os dois sockets.
- ❑ Código de exemplo para fechar um socket:

```
int sd; // socketdescriptor
...

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

© 2004-2015 Volnys Bernal 35

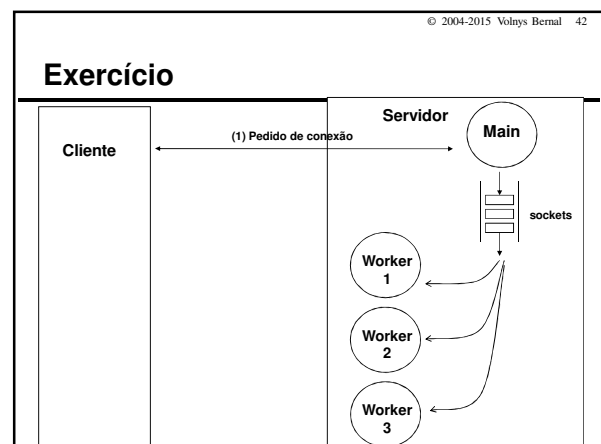
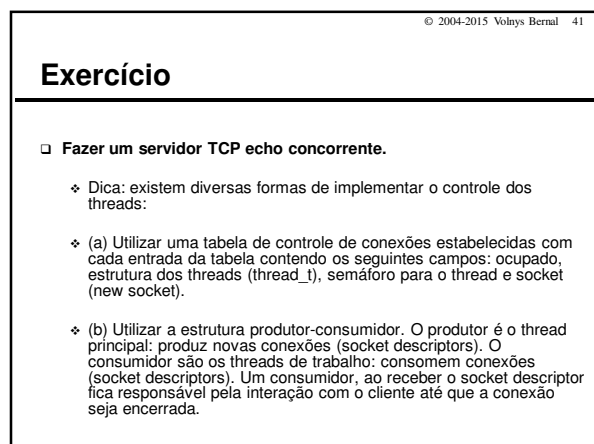
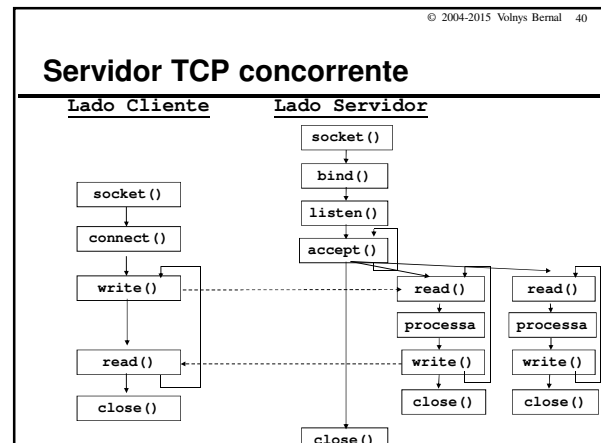
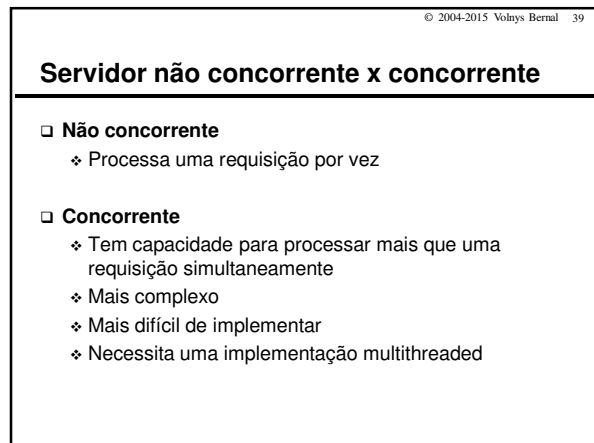
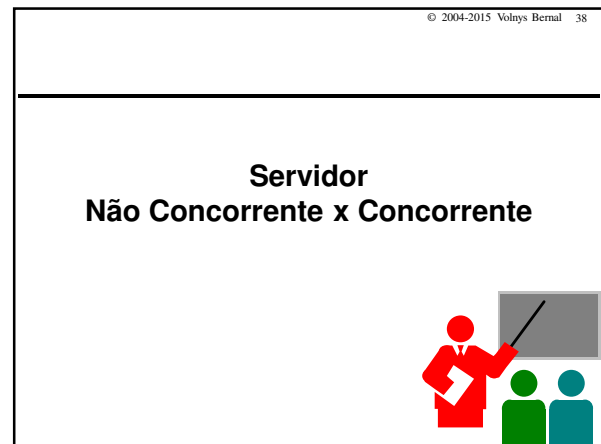
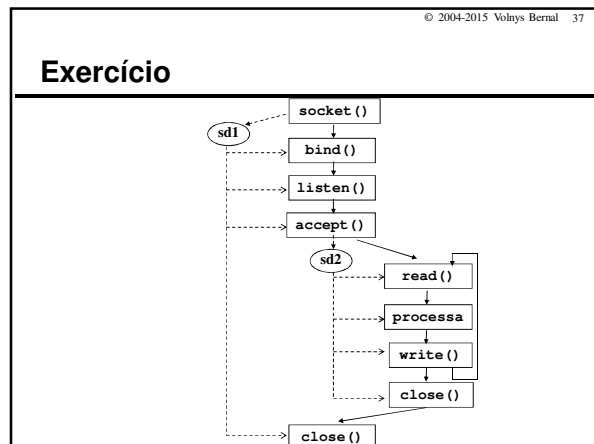
Exercício

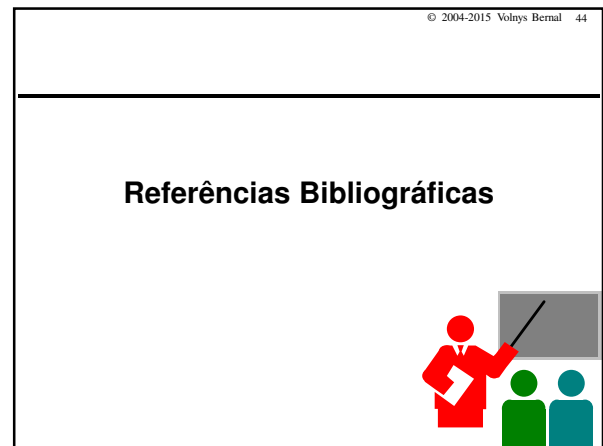
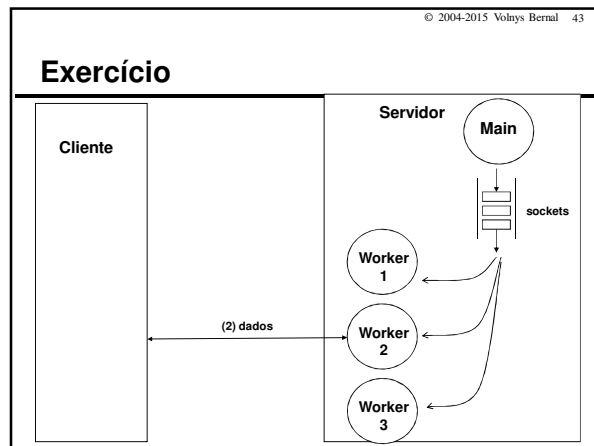
© 2004-2015 Volnys Bernal 36

Exercício

1. **Implemente um servidor TCP echo que transforma para maiúsculas.**
 - ❖ O programa deve atender a um cliente TCP e, quando receber a mensagem "quit" deve encerrar a conexão com o cliente e terminar o programa
 - ❖ Dicas:
 - Utilize a rotina de biblioteca `toupper()` para converter um caractere minúsculo para maiúsculo.
 - Utilize a rotina de biblioteca `strcmp()` para comparar a string recebida com "quit".

```
#include <string.h>
If ( strcmp(bufferp, "quit")==0)
    /* igual */
```





© 2004-2015 Volnys Bernal 45

Referências Bibliográficas

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