“um socket é a combinação de um endereço IP, um protocolo e o número da porta do protocolo”

Material de apoio utilizado nas aulas de TCD – Tecnologia de Comunicação de Dados
IST – Instituto Superior de Tecnologia – Petrópolis
Prof. Luis Rodrigo de O. Goncalves
luisrodrigoog@gmail.com

Atualizado em : 23/08/07 - Impresso em : 23/08/07

Sockets - Conceitos:

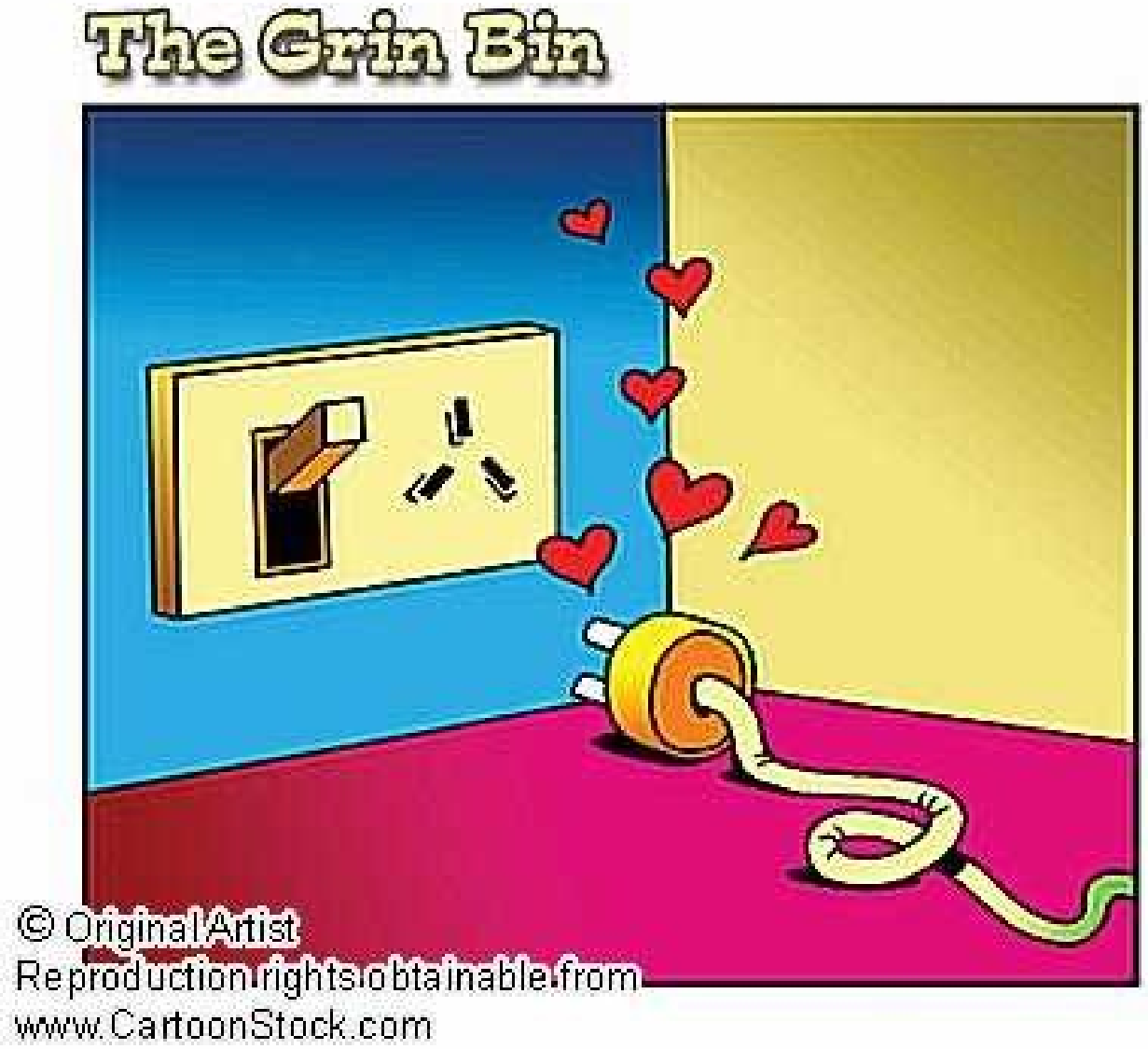
- 1ª Implementação: Unix BSD;
- Berkeley Sockets;
- Possibilita a comunicação bi-direcional;
- Esconde os detalhes de baixo nível;
- Baseados nos descritores de arquivo;
- Chamadas: send () e recv ();
- Tipos Básicos: DARPA, Socket Unix, X.25.
- Internet Sockets: Stream Sockets,
  Datagram Sockets
  e Raw Sockets

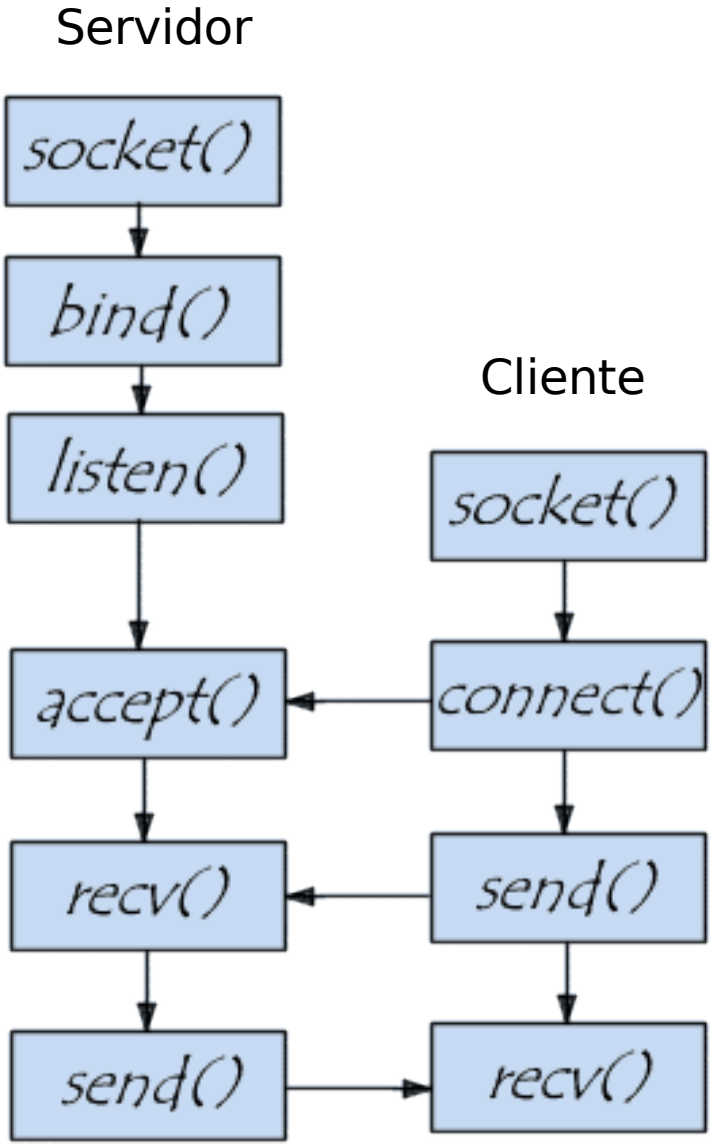
Prof. Luis Rodrigo
luisrodrigoog@gmail.com
http:// www.lncc.br/~lrodrigo

Sockets - Conceitos:

- Datagram Sockets (UDP):
    - Sem conexão;
    - Transferência pacote por pacote;
    - Mais rápido;
- Stream Sockets (TCP) :
    - Com conexão;
    - Fluxo confiável de dados;
    - Envio de uma grade quantidade de dados;
- Raw Sockets :
    - permite acesso aos protocolos de Rede;

Prof. Luis Rodrigo
luisrodrigoog@gmail.com
http:// www.lncc.br/~lrodrigo





Prof. Luis Rodrigo
luisrodrigoog@gmail.com
http:// www.lncc.br/~lrodrigo

Cabeçalhos Básicos :

```
#include <string.h>          //String operations
#include <stdio.h>           //Standard buffered input/output
#include <stdlib.h>          //Standard library definitions
#include <errno.h>           //System error numbers
#include <netdb.h>           //Network database operations
#include <sys/types.h>       //Data types
#include <sys/socket.h>      //Internet Protocol family(access)
 #include <netinet/in.h>     //Internet Protocol family (types)
 #include <arpa/inet.h>      //Definitions for internet operation
```

Mais informações clique Aqui

Prof. Luis Rodrigo
luisrodrigoog@gmail.com
http:// www.lncc.br/~lrodrigo

Estruturas de Dados :

```
struct sockaddr {
        unsigned short sa_family;  // tipo do endereço
        char sa_data [14];         //  dados do endereçamento
}
```

sa_family: AF_INET

AF_UNIX

AF_NS

AF_ISO

Prof. Luis Rodrigo
luisrodrigoog@gmail.com
http:// www.lncc.br/~lrodrigo

Estruturas de Dados:

```
struct sockaddr_in {
        unsigned short sa_family;      // tipo do endereço
        unsigned short int sin_port;   // numero da porta
        struct in_addr sin_addr        // end. da Internet
        unsigned char sin_zero[8]      // manter o  padrão
}

struct in_addr {
    unsigned long s_addr        // endereço IPv4
}
```

Prof. Luis Rodrigo
luisrodrigoog@gmail.com
http:// www.lncc.br/~lrodrigo

Ordenações de Bytes:

- Network Byte Order – Little Endian (1º bit menos significativo)
- Host Byte Order – Big Endian (1º bit mais significativo)

Utiliza-se por padrão o Host Byte Order;

Campos sin_port, sin_addr e sin_addr.s_addr :

- são encapsulados;
- devem ser convertidos para Network Byte Order;

Prof. Luis Rodrigo
luisrodrigoog@gmail.com
http:// www.lncc.br/~lrodrigo

Convertendo Valores:

- htons ( ) - Host to Network Short;
- htonl ( ) - Host to Network Long;
- ntohs ( ) - Network to Host Short;
- ntohl ( ) - Network to Host Long;
- inet_addr (“end”) - End. IP em unsinged long;
- inet_aton (“end”, “varConvertida”) - ASCII to Network;
- inet_ntoa ( ) - Network to ASCII.

Prof. Luis Rodrigo
luisrodrigoog@gmail.com
http:// www.lncc.br/~lrodrigo

Exemplo 01: estrutura de endereçamento

```
struct sockaddr_in  endLocal;
endLocal.sin_family=AF_INET;
endLocal.sin_port=htons(1502);
inet_aton("10.15.02.94",
                          &(endLocal.sin_addr));
memset(&(endLocal.sin_zero),'\0',8);
```

Prof. Luis Rodrigo
luisrodrigoog@gmail.com
http:// www.lncc.br/~lrodrigo

Exemplo 02: Convertendo endereços

```
struct sockaddr_in end1, end2;
char *a1;
char *a2;
inet_aton("10.15.02.94", &(end1));
inet_aton("10.15.02.99", &(end2));
a1=inet_ntoa(end1);
a2=inet_ntoa(end2);
printf ("Endereço 1: %s", a1);
printf ("Endereço 2: %s", a2);
```

Prof. Luis Rodrigo
luisrodrigoog@gmail.com
http:// www.lncc.br/~lrodrigo

Chamadas de Sistema: socket ( )

- Cria do descritor associado ao socket;

int socket (int dominio, int tipo, int protocol);

dominio: AF_INET;

tipo : SOCK_STREAM, SOCK_DGRAM;

protocol: 0;

Retorna:

"-1" em caso de erro e seta a variável "errno" ;

Prof. Luis Rodrigo
luisrodrigoog@gmail.com
http:// www.lncc.br/~lrodrigo

Chamadas de Sistema: bind ( )

Associa o socket a uma porta;

Permite ao kernel enviar um pacote a um processo;

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

sockfd = descritor do socket;

endLocal = ponteiro para uma estrutura as informações do socket

addrlen = sizeof (struct sockaddr);

Prof. Luis Rodrigo
luisrodrigoog@gmail.com
http:// www.lncc.br/~lrodrigo

Exemplo 03: Utilizando o bind ()

```
main ( ){
    int sockServ;
    struct sockaddr_in endLocal;
    sockServ= socket (AF_INET,SOCK_STREAM,0);
    endLocal.sin_family=AF_INET;
    endLocal.sin_port=htons(3490);
    endLocal.sin_addr.s_addr=inet_addr("127.0.0.1");
    memset(&(endLocal.sin_zero),'\0',8);
    bind(sockServer, (struct sockaddr *)&endLocal,
                                    sizeof(struct sockaddr));
}
```

Prof. Luis Rodrigo
luisrodrigoog@gmail.com
http:// www.lncc.br/~lrodrigo

Obtendo IP e Porta:

```
endLocal.sin_port=0;
endLocal.sin_addr.s_addr=INADDR_ANY;
```

ou

```
endLocal.sin_port=htons(0);
endLocal.sin_addr.s_addr=htonl(INADDR_ANY);
```

Escolha das Portas :

$$1024 < sin\_port < 65536$$

Consulte: /etc/services

Prof. Luis Rodrigo
luisrodrigoog@gmail.com
http:// www.lncc.br/~lrodrigo

Address already in use:

```
int pos = 1;
// anula a mesagem ``Address already in use''
if  (setsockopt(listener,SOL_SOCKET,
                SO_REUSEADDR,&pos,sizeof(int)) == -1)
{
    perror(``setsockopt'');
    exit(1);
}
```

Prof. Luis Rodrigo
luisrodrigoog@gmail.com
http:// www.lncc.br/~lrodrigo

Chamadas de Sistema: conect ( )

- realiza a conexão do socket local a um remoto;

```
int connect (int sockfd, struct sockaddr
                        *servAddr, int addrlen);
```

sockfd = descritor do socket;

servAddr = ponteiro para uma estrutura com o endereço do servidor

addrlen = sizeof (struct sockaddr);

Prof. Luis Rodrigo
luisrodrigoog@gmail.com
http:// www.lncc.br/~lrodrigo

Exemplo: clienteParcial-01.c

```
main() {
    int sockCli;
    struct sockaddr_in end_dest;
    sockCli = socket(AF_INET, SOCK_STREAM, 0);
    end_dest.sin_family = AF_INET;
    end_dest.sin_port = htons(22);
    end_dest.sin_addr.s_addr = inet_addr("146.134.12.1");
    memset(&(end_dest.sin_zero), '\0', 8);
    connect(sockCli, (struct sockaddr *)&end_dest,
                                    sizeof(struct sockaddr));
... }
```

Prof. Luis Rodrigo
luisrodrigoog@gmail.com
http:// www.lncc.br/~lrodrigo

Chamadas de Sistema: listen ( )

- permite ao processo escutar em uma porta;
- permite determinar a chegada de solicitações;

```
int listen( int sockfd, int backLog);
```

sockfd = descritor do socket;

backLog = limite de quantas conexões que ficarão na fila, geralmente 20;

Prof. Luis Rodrigo
luisrodrigoog@gmail.com
http:// www.lncc.br/~lrodrigo

Chamadas de Sistema: accept ( )

- pega as conexões da fila gerada pelo listen ();
- obtém um novo descritor a ser utilizado na transferência de dados;

```
int accept (int sockfd, void *addrCli,
                        int *addrLen);
```

sockfd = descritor do socket;

addrCli = estrutura de endereçamento (sockaddr_in.) com os dados do cliente;

addrLen = sizeof(struct sockaddr_in);

Prof. Luis Rodrigo
luisrodrigoog@gmail.com
http:// www.lncc.br/~lrodrigo

Exemplo: accept.c

```
main() {
    int sockServer, sockCli;
    struct sockaddr_in endServer, endCli;
    int sin_size;
    sockServer = socket(AF_INET, SOCK_STREAM, 0);
    endServer.sin_family = AF_INET;
    endServer.sin_port = htons(3490);
    endServer.sin_addr.s_addr = INADDR_ANY;
    memset(&(endServer.sin_zero), '\0', 8);
    ...
    ...
    ...
```

Prof. Luis Rodrigo
luisrodrigoog@gmail.com
http:// www.lncc.br/~lrodrigo

Exemplo: accept.c

```
    ...
    ...
    bind (sockServer, (struct sockaddr *)&endServer,
                                        sizeof(struct sockaddr));
    listen(sockServer, 10 );
    sin_size = sizeof(struct sockaddr_in);
    sockCli = accept(sockServer, &endCli, &sin_size);
}
```

Prof. Luis Rodrigo
luisrodrigoog@gmail.com
http:// www.lncc.br/~lrodrigo

Chamadas de Sistema: send ()

- envia bytes através de um SOCK_STREAM
- retorna a quantidade de bytes enviados;

```
int send(int sockfd, const void *msg,
                                int len, int flags);
```

*msg = dados a serem enviados;

len = qt de bytes a serem enviados;

flags = 0.

Prof. Luis Rodrigo
luisrodrigoog@gmail.com
http:// www.lncc.br/~lrodrigo

Exemplo: send.c

```
main() {
    ...
    ...
    char *msg = "Oi tudo bem?";
    int tamanho, bytes_enviados;
    ...
    tamanho = strlen(msg);
    bytes_enviados = send(sockCli, msg, tamanho, 0);
    ...
    ...
}
```

Prof. Luis Rodrigo
luisrodrigoog@gmail.com
http:// www.lncc.br/~lrodrigo

Chamadas de Sistema: recv ()

- permite receber bytes através de um SOCK_STREAM
- retorna a quantidade de bytes recebidos

```
int recv(int sockfd, void *buf, int len,
                                unsigned int flags);
```

***buf** = ponteiro para a variável que receberá os dados ;

len = tamanho do *buffer*;

flags = 0.

Prof. Luis Rodrigo
luisrodrigoog@gmail.com
http:// www.lncc.br/~lrodrigo

Chamadas de Sistema: close ()

- fecha uma determinada conexão;

```
close(sockfd);
```

Chamadas de Sistema: shutdown ()

- altera a “usabilidade” do socket;

```
int shutdown(int sockfd, int how);
```

- how pode assumir:

0 – encerra recebimento

1 – encerra o envio

2 – encerra recebimento e envio

Prof. Luis Rodrigo
luisrodrigoog@gmail.com
http:// www.lncc.br/~lrodrigo

Chamadas de Sistema: getpeername ( )

- quem está conectado na outra extremidade;

```
int getpeername(int sockfd, struct
                sockaddr *addr, int *addrlen);
```

addr = dados sobre a maquina remota;

addrlen = sizeof(struct sockaddr);

Prof. Luis Rodrigo
luisrodrigoog@gmail.com
http:// www.lncc.br/~lrodrigo

Chamadas de Sistema: gethostname ( )

- nome do nó onde o programa está rodando;

```
int gethostname (char *hostname,
                 size_t size);
```

hostname = nome da maquina;

size = tamanho do nome da maquina;

Prof. Luis Rodrigo
luisrodrigoog@gmail.com
http:// www.lncc.br/~lrodrigo

Chamadas de Sistema: gethostbyname ( )

- obtem os dados (IP/FQDN) de uma maquina;

```
struct hostent *gethostbyname(
                        const char *name);

struct hostent {
        char *h_name;       // nome oficial
        char **h_aliases;   // alias (vetor)
        int h_addrtype;     // tipo do endereço
        int h_length;       // tamanho do endereço
        char **h_addr_list; // endereços IP (vetor)
    }
```

Prof. Luis Rodrigo
luisrodrigoog@gmail.com
http:// www.lncc.br/~lrodrigo

Exemplo: pegaIP.c

```
int main(int argc, char *argv[]){
        struct hostent *h;
        if (argc != 2) {
              fprintf(stderr, "Uso: getip endereço\n");
              exit(1);
        }

        if ((h=gethostbyname(argv[1])) == NULL)  {
               herror("gethostbyname");
               exit(1);
        }
        ...
        ...
```

Prof. Luis Rodrigo
luisrodrigoog@gmail.com
http:// www.lncc.br/~lrodrigo

Exemplo: pegaIP.c

```
        ...
        ...
        printf ("Nome do Host : %s\n", h->h_name);
        printf ("Endereço IP  : %s\n",
        inet_ntoa(*((struct in_addr *)h->h_addr_list[0] )));
        return 0;
}
```

Prof. Luis Rodrigo
luisrodrigoog@gmail.com
http:// www.lncc.br/~lrodrigo

Exemplo: scannerDePortas.c (1)

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <sys/socket.h>
#include <sys/types.h>
#include <netinet/in.h>
#include <sys/signal.h>
#include <errno.h>
#include <netdb.h>
#define PORTA_INICIO 1
```

Prof. Luis Rodrigo
luisrodrigoog@gmail.com
http:// www.lncc.br/~lrodrigo

Exemplo: scannerDePortas.c (2)

```
main(int argc, char *argv[])
{
 char host[15];
 int porta, portas, i , spawnsocket, delay, ligacao;
 struct sockaddr_in alvo;
 struct hostent *he;
 struct servent *servicos;


     if ( argc == 1 ){
         fprintf(stderr,"Scanner de Portas TCP Abertas\n");
             fprintf(stderr,"usar: %s <host>\n",argv[0]);
         exit(0);
     }
```

Prof. Luis Rodrigo
luisrodrigoog@gmail.com
http:// www.lncc.br/~lrodrigo

Exemplo: scannerDePortas.c (3)

```
if ( argc > 1 ) porta = PORTA_INICIO;
if (argc  > 2 ) porta = atoi (argv[2]);
// obtem dados da maquina ( nome & endereço (IP) )
he = gethostbyname ( argv[1] );
if ( he == NULL ) {
    printf("Host Nao encontrado \n");
    exit(-1);
}
for(porta=1;porta<=1024;porta++)   {
spawnsocket = socket( AF_INET, SOCK_STREAM, 0);
if ( spawnsocket < 0)
        perror("Socket");
```

Prof. Luis Rodrigo
luisrodrigoog@gmail.com
http:// www.lncc.br/~lrodrigo

Exemplo: scannerDePortas.c (4)

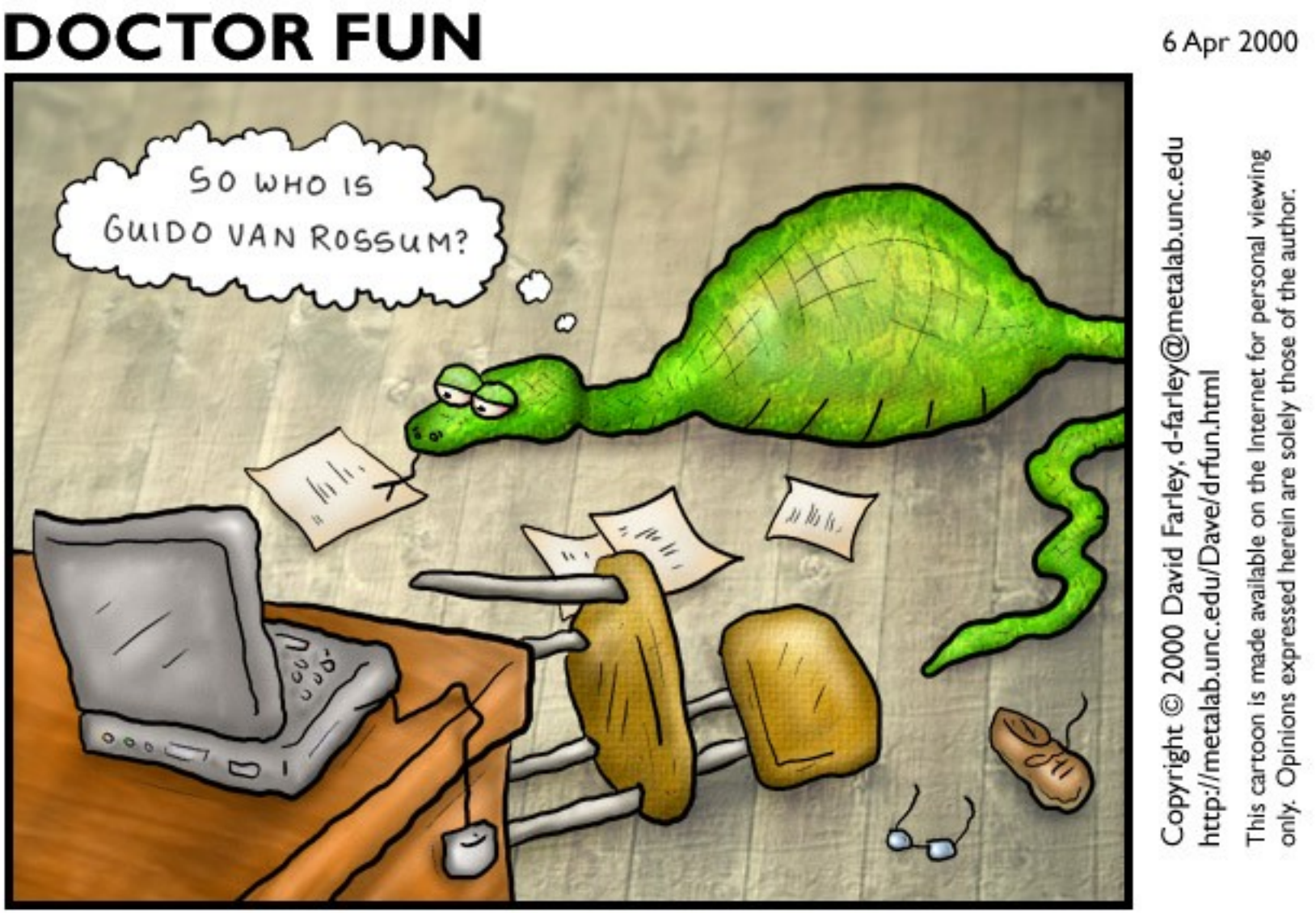
```
alvo.sin_family = he->h_addrtype;
alvo.sin_port = htons(porta);
alvo.sin_addr = *((struct in_addr *)he->h_addr);
bzero(&(alvo.sin_zero),8);
ligacao = connect ( spawnsocket, (struct sockaddr *) &alvo,
sizeof(alvo));
if ( ligacao == -1 ) {
    // perror("Porta esta fechada\n");
    //printf("A porta %d esta fechada\n",porta);
}
```

Prof. Luis Rodrigo
luisrodrigoog@gmail.com
http:// www.lncc.br/~lrodrigo

Exemplo: scannerDePortas.c (5)

```
        else {
            printf("A porta %d esta ABERTA\n",porta);
        }
        close(spawnsocket);
    }
    printf ("\n");
}
```

Prof. Luis Rodrigo
luisrodrigoog@gmail.com
http:// www.lncc.br/~lrodrigo


DOCTOR FUN
6 Apr 2000
SO WHO IS GUIDO VAN ROSSUM?
Copyright © 2000 David Farley, d-farley@metalab.unc.edu
http://metalab.unc.edu/Dave/drfun.html
This cartoon is made available on the Internet for personal viewing only. Opinions expressed herein are solely those of the author.

Módulos Necessários:

```
// obtendo acesso a interface BSD Sockets
import socket
// obtendo acesso a interface do sistema
import sys
```

Prof. Luis Rodrigo
luisrodrigoog@gmail.com
http:// www.lncc.br/~lrodrigo

Criando o descritor:

dsSocket = socket.socket (family, type)

onde:

dsSocket : contem o valor do descritor;

family: AF_UNIX / AF_INET

type: SOCK_STREAM, SOCK_DATAGRAM

```
dsSocket= socket.socket (socket.AF_INET,
                         socket.SOCK_STREAM)
```

Prof. Luis Rodrigo
luisrodrigoog@gmail.com
http:// www.lncc.br/~lrodrigo

Definindo o endereço:

host="127.0.0.1"

porta="4935"

descEnd = (host , porta)

onde:

host : endereço IP da máquina

porta: porta onde ocorrerá as conexões

descEnd: é a tupla que identifica o endereço

Prof. Luis Rodrigo
luisrodrigoog@gmail.com
http:// www.lncc.br/~lrodrigo

Associando a porta ao processo:

descSock.bind (descEnd)

onde:

descSock : é o descritor criado anteriormente

descEnd: é a tupla que identifica o endereço

endIP="127.0.0.1"

porta="4935"

endServidor (endIP, porta)

sockServidor.bind (endServidor)

Prof. Luis Rodrigo
luisrodrigoog@gmail.com
http:// www.lncc.br/~lrodrigo

Escutando a porta:

descSock.listen (backlog)

onde:

backLog : número máximo de conexões da fila de espera;

sockServidor.listen (1)

Prof. Luis Rodrigo
luisrodrigoog@gmail.com
http:// www.lncc.br/~lrodrigo

Aceitando Conexões

sockCliente , endCliente = descSock.accept ( )

onde:

sockCliente : novo socket associado ao cliente;

endCliente : end. remoto do socket cliente;

Prof. Luis Rodrigo
luisrodrigoog@gmail.com
http:// www.lncc.br/~lrodrigo

Conectando-se ao Servidor

descSocket.connect (endServidor)

onde:

descSocket : descritor anteriormente criado;

endservidor : tupla com os dados do endereço do servidor;

sockCliente.connect (endServidor)

Prof. Luis Rodrigo
luisrodrigoog@gmail.com
http:// www.lncc.br/~lrodrigo

Fechando a conexão:

descSocket.close ( )

onde:

descSocket : descritor do socket a ser encerrado;

descSocket.shutdown (como)

onde:

como : 0 – impede o recebimento;

1 – impede o envio;

2 – impede o envio e o recebimento.

Prof. Luis Rodrigo
luisrodrigoog@gmail.com
http:// www.lncc.br/~lrodrigo

Enviando dados:

qtBytes = descSocket.send (mensagem)

onde:

descSocket : descritor anteriormente criado;

mensagen : conteúdo da msg a ser enviada;

qtBytes : quantidade de bytes enviados;

msg= “Olá, mundo !!! \n ”

bytesEnviados = sockCliente.send (msg)

Prof. Luis Rodrigo
luisrodrigoog@gmail.com
http:// www.lncc.br/~lrodrigo

Recebendo dados:

mensagem = descSocket.recv (qyBytes)

onde:

descSocket : descritor anteriormente criado;

mensagem : conteúdo da mensagem recebida;

qtBytes : quantidade de bytes recebidos;

```
msg = sockCliente.recv ( 1024 )
print msg
```

Prof. Luis Rodrigo
luisrodrigoog@gmail.com
http:// www.lncc.br/~lrodrigo

Exemplo : servidorSimples.py

```
# importando modulos
import socket
# definindo o endereço
HOST = ''                 # Endereco IP do Servidor
PORT = 5000               # Porta que o Servidor esta
# criando o descritor
tcp = socket.socket(socket.AF_INET,
                    socket.SOCK_STREAM)
orig = (HOST, PORT)
# associando o processo a porta
tcp.bind(orig)
# escutando a porta
tcp.listen(1)
```

Prof. Luis Rodrigo
luisrodrigoog@gmail.com
http:// www.lncc.br/~lrodrigo

Exemplo : servidorSimples.py

```
# laço principal
while True:
    # recebe uma nova conexão
    con, cliente = tcp.accept()
    print 'Concetado por', cliente
    # laço para envio e recebimento de dados
    while True:
        msg = con.recv(1024)
        if not msg: break
        print cliente, msg
    print 'Finalizando conexao do cliente', cliente
    # encerra o socket
    con.close()
```

Prof. Luis Rodrigo
luisrodrigoog@gmail.com
http:// www.lncc.br/~lrodrigo

Exemplo : servidorConcorrente.py

```
# Importando os módulos
import socket
import os
import sys
# Definindo o endereço
HOST = ''        # Endereco IP do Servidor
PORT = 5000   # Porta que o Servidor esta
# Criando o socket
tcp = socket.socket(socket.AF_INET,
                    socket.SOCK_STREAM)
```

Prof. Luis Rodrigo
luisrodrigoog@gmail.com
http:// www.lncc.br/~lrodrigo

Exemplo : servidorConcorrente.py

```
...
# Associando a Porta do Processo
orig = (HOST, PORT)
tcp.bind(orig)
# Aguardando por conexões
tcp.listen(1)
#  Laço principal
while True:
        # aceitando conexões
        con, cliente = tcp.accept()
```

Prof. Luis Rodrigo
luisrodrigoog@gmail.com
http:// www.lncc.br/~lrodrigo

Exemplo : servidorConcorrente.py

```
...     ...
        #  gera um novo processo
        pid = os.fork()
        # verifica se é o cliente
        if pid == 0:
            # fecha o socket do proceso Pai
            tcp.close()
           # imprime dados do cliente
           print 'Conectado por, cliente
           # trata a nova conexão
          while True:
                  # recebe dados
                 msg = con.recv(1024)
                  # verifica se recebeu algo
                if not msg: break
                 # imprime dados
               print cliente, msg
```

Prof. Luis Rodrigo
luisrodrigoog@gmail.com
http:// www.lncc.br/~lrodrigo

Exemplo : servidorConcorrente.py

```
            ...
        print 'Finalizando conexao do cliente', cliente
        # finaliza conexão
        con.close()
        # encerra o processo
        sys.exit(0)
    else:
        # se for o pai, fecha o socket filho
        con.close()
```

Prof. Luis Rodrigo
luisrodrigoog@gmail.com
http:// www.lncc.br/~lrodrigo

Exemplo : servidorThreads.py

```
# importando os modulos
import socket
import thread
# dados do endereço
HOST = ''
PORT = 5000
# funcao para tratar das conexões
def conectado(con, cliente):
    print 'Conectado por', cliente
    # laço para tratar a coexão
    while True:
        msg = con.recv(1024)
        if not msg: break
        print cliente, msg
```

Prof. Luis Rodrigo
luisrodrigoog@gmail.com
http:// www.lncc.br/~lrodrigo

Exemplo : servidorThreads.py

```
    print 'Finalizando conexao do cliente', cliente
    con.close()
    thread.exit()

# corpo do programa
tcp = socket.socket(socket.AF_INET,
                                    socket.SOCK_STREAM)
# formando o endereço
orig = (HOST, PORT)
# associando o processo a porta
tcp.bind(orig)
```

Prof. Luis Rodrigo
luisrodrigoog@gmail.com
http:// www.lncc.br/~lrodrigo

Exemplo : servidorThreads.py

```
# passa a escutar a porta
tcp.listen(1)

# laço principal do programa
while True:
    # aceita novas coneções
    con, cliente = tcp.accept()
    # cria uma nova thread para tratar a conexão
    thread.start_new_thread(conectado,
                            tuple([con, cliente]))

tcp.close()
```

Prof. Luis Rodrigo
luisrodrigoog@gmail.com
http:// www.lncc.br/~lrodrigo

Exemplo : cliente.py

```
# importa o modelo
import socket
# define o endereço do servidor
HOST = '127.0.0.1'
PORT = 5000
# cria o descritor
tcp = socket.socket(socket.AF_INET,
                    socket.SOCK_STREAM)
dest = (HOST, PORT)
# conecta ao servidor
tcp.connect(dest)
print 'Para sair use CTRL+X\n'
```

Prof. Luis Rodrigo
luisrodrigoog@gmail.com
http:// www.lncc.br/~lrodrigo

Exemplo : cliente.py

```
# obtem os dados a serem enviados
msg = raw_input()
# verifica se deve encerrar ou não
while msg <> '\x18':
    # envia os dados
    tcp.send (msg)
    # obtem novos dados
    msg = raw_input()
# fecha a conexão com o servidor
tcp.close()
```

Prof. Luis Rodrigo
luisrodrigoog@gmail.com
http:// www.lncc.br/~lrodrigo

Sockets - Referencias:

Python:

[1] http://www.pythonbrasil.com.br/moin.cgi/CookBook

[2] http://docs.python.org/

[3] http://www.inf.ufrgs.br/~psgrigoletti/docs/artigo_funcionalidades_python.pdf

[4] http://www-users.cs.york.ac.uk/~aw/pylinda/beginner.html

[5] http://www-users.cs.york.ac.uk/~aw/pylinda/beginner.html

[6] http://www.pyzine.com/

[7] http://aspn.activestate.com/ASPN/Cookbook/Python/

[8] http://pt.wikipedia.org/wiki/Python

[9] http://www.onlamp.com/python/

[10] http://py.vaults.ca/apyllo.py

Prof. Luis Rodrigo
luisrodrigoog@gmail.com
http:// www.lncc.br/~lrodrigo