



Exercício 2

Aluno: Caio Augusto Alves Nolasco

RA: 195181

Instituto de Computação

Universidade Estadual de Campinas

Campinas, 24 de Novembro de 2020.

Sumário

1	Questão 1	2
2	Questão 2	2
3	Questão 3	3
4	Questão 4	6

1 Questão 1

De maneira geral, o backlog do socket do servidor informa o número máximo de conexões pendentes suportadas por um socket TCP. De forma mais específica, o tamanho definido pelo valor do parâmetro backlog da função `listen()` define o valor máximo da soma dos tamanhos de duas filas de conexões no socket: uma fila para conexões que ainda esperam o término do three way handshake e outra fila para conexões que já tiveram o handshake concluído.

2 Questão 2

Para kernels do Linux 2.2 em diante, o valor de backlog especifica o tamanho máximo de file para conexões estabelecidas que esperam ser aceitas. O valor padrão de backlog é salvo em uma variável de sistema no endereço `/proc/sys/net/core/somaxconn`, com seu valor padrão igual a 128.

A seguir, uma foto da manpage da chamada `listen()`

```
File Edit View Search Terminal Help
cnoiasco@calopc: ~
EBADF The argument sockfd is not a valid file descriptor.

ENOTSOCK
The file descriptor sockfd does not refer to a socket.

EOPNOTSUPP
The socket is not of a type that supports the listen() operation.

CONFORMING TO
POSIX.1-2001, POSIX.1-2008, 4.4BSD (listen() first appeared in 4.2BSD).

NOTES
To accept connections, the following steps are performed:

1. A socket is created with socket(2).
2. The socket is bound to a local address using bind(2), so that other sockets may be connect(2)ed to it.
3. A willingness to accept incoming connections and a queue limit for incoming connections are specified with listen().
4. Connections are accepted with accept(2).

POSIX.1 does not require the inclusion of <sys/types.h>, and this header file is not required on Linux. However, some historical (BSD) implementations required this header file, and portable applications are probably wise to include it.

The behavior of the backlog argument on TCP sockets changed with Linux 2.2. Now it specifies the queue length for completely established sockets waiting to be accepted, instead of the number of incomplete connection requests. The maximum length of the queue for incomplete sockets can be set using /proc/sys/net/ipv4/tcp_max_syn_backlog. When syncookies are enabled there is no logical maximum length and this setting is ignored. See tcp(7) for more information.

If the backlog argument is greater than the value in /proc/sys/net/core/somaxconn, then it is silently truncated to that value; the default value in this file is 128. In kernels before 2.4.25, this limit was a hard coded value, SOMAXCONN, with the value 128.

EXAMPLE
See bind(2).

SEE ALSO
accept(2), bind(2), connect(2), socket(2), socket(7)

COLOPHON
Manual page listen(2) line 43/99 93% (press h for help or q to quit)
```

3 Questão 3

Os dez clientes são conectados simultaneamente usando o programa Terminator para executar o cliente. Simultaneamente, o comando netstat pertinente para a porta do servidor também é chamado nesse momento.


```
cnolasco@calopc:~/Documents/Ex2-MC833/Ex3$  
cnolasco@calopc:~/Documents/Ex2-MC833/Ex3$  
cnolasco@calopc:~/Documents/Ex2-MC833/Ex3$ sudo netstat -tulpn | grep 33333  
tcp        2      0 0.0.0.0:33333        0.0.0.0:*            LISTEN      25788/.servidor  
tcp        0      0 127.0.0.1:57570      127.0.0.1:33333      SYN_SENT    25714/.cliente  
tcp        0      0 127.0.0.1:57562      127.0.0.1:33333      ESTABLISHED 25710/.cliente  
tcp        0      0 127.0.0.1:57572      127.0.0.1:33333      SYN_SENT    25715/.cliente  
tcp        0      0 127.0.0.1:57576      127.0.0.1:33333      SYN_SENT    25717/.cliente  
tcp        0      0 127.0.0.1:33333      127.0.0.1:57560      ESTABLISHED -  
tcp        0      0 127.0.0.1:57566      127.0.0.1:33333      SYN_SENT    25712/.cliente  
tcp        0      0 127.0.0.1:33333      127.0.0.1:57562      ESTABLISHED -  
tcp        0      0 127.0.0.1:57574      127.0.0.1:33333      SYN_SENT    25716/.cliente  
tcp        0      0 127.0.0.1:57578      127.0.0.1:33333      SYN_SENT    25718/.cliente  
tcp        0      0 127.0.0.1:57564      127.0.0.1:33333      SYN_SENT    25711/.cliente  
tcp        0      0 127.0.0.1:57568      127.0.0.1:33333      ESTABLISHED 25709/.cliente  
tcp        0      0 127.0.0.1:57568      127.0.0.1:33333      SYN_SENT    25713/.cliente  
cnolasco@calopc:~/Documents/Ex2-MC833/Ex3$
```

Com o backlog setado para 9, todos os cliente conseguem se conectar instantaneamente ao servidor.

```
cnolasco@calopc:~/Documents/Ex2-MC833/Ex3$  
cnolasco@calopc:~/Documents/Ex2-MC833/Ex3$  
cnolasco@calopc:~/Documents/Ex2-MC833/Ex3$ netstat -tulpn | grep 33333  
(net all processes could be identified, non-owned process info  
will not be shown, you would have to be root to see it all.)  
cnolasco@calopc:~/Documents/Ex2-MC833/Ex3$ netstat -tulpn | grep 33333  
(net all processes could be identified, non-owned process info  
will not be shown, you would have to be root to see it all.)  
tcp        10      0 0.0.0.0:33333        0.0.0.0:*            LISTEN      26095/.servidor  
tcp        0      0 127.0.0.1:57778      127.0.0.1:33333      ESTABLISHED 26104/.cliente  
tcp        0      0 127.0.0.1:57770      127.0.0.1:33333      ESTABLISHED 26100/.cliente  
tcp        0      0 127.0.0.1:33333      127.0.0.1:57778      ESTABLISHED -  
tcp        0      0 127.0.0.1:57766      127.0.0.1:33333      ESTABLISHED 26098/.cliente  
tcp        0      0 127.0.0.1:33333      127.0.0.1:57766      ESTABLISHED -  
tcp        0      0 127.0.0.1:57780      127.0.0.1:33333      ESTABLISHED 26105/.cliente  
tcp        0      0 127.0.0.1:33333      127.0.0.1:57764      ESTABLISHED -  
tcp        0      0 127.0.0.1:57772      127.0.0.1:33333      ESTABLISHED 26101/.cliente  
tcp        0      0 127.0.0.1:57768      127.0.0.1:33333      ESTABLISHED 26099/.cliente  
tcp        0      0 127.0.0.1:33333      127.0.0.1:57780      ESTABLISHED -  
tcp        0      0 127.0.0.1:33333      127.0.0.1:57778      ESTABLISHED -  
tcp        0      0 127.0.0.1:57762      127.0.0.1:33333      ESTABLISHED 26096/.cliente  
tcp        0      0 127.0.0.1:33333      127.0.0.1:57774      ESTABLISHED -  
tcp        0      0 127.0.0.1:33333      127.0.0.1:57772      ESTABLISHED -  
tcp        0      0 127.0.0.1:57764      127.0.0.1:33333      ESTABLISHED 26097/.cliente  
tcp        0      0 127.0.0.1:57776      127.0.0.1:33333      ESTABLISHED 26103/.cliente  
tcp        0      0 127.0.0.1:33333      127.0.0.1:57768      ESTABLISHED -  
tcp        0      0 127.0.0.1:33333      127.0.0.1:57762      ESTABLISHED -  
tcp        0      0 127.0.0.1:57774      127.0.0.1:33333      ESTABLISHED 26102/.cliente  
cnolasco@calopc:~/Documents/Ex2-MC833/Ex3$
```

4 Questão 4

O programa atualmente não faz uma chamada a função `wait()` e nem a função `waitpid()`, logo os processos filhos se tornam zumbis. Para lidar com os filhos zumbis, faz-se uma chamada para reconhecer o sinal `SIGCHLD`, e então uma chamada para `wait()`.

```
void sig_chld() {  
    pid_t pid;  
    int stat;  
  
    pid = wait(&stat);  
    printf("child %d terminated\n", pid);  
}
```

```
bzero(&servaddr, sizeof(servaddr));  
servaddr.sin_family = AF_INET;  
servaddr.sin_addr.s_addr = htonl(0); //Adota como endereço de domínio o IP da máquina do servidor  
servaddr.sin_port = htons(atoi(argv[1])); //Permite que o sistema atribua um número de porta temporário  
  
Bind(listenfd, (struct sockaddr *)&servaddr, sizeof(servaddr)); //Unir o socket com a porta de número especificada  
Listen(listenfd, 9); //Iniciar a escuta por conexões, usando o descritor de socket listenfd criado  
  
Signal(SIGCHLD, sig_chld);  
  
for (;;) {  
    sleep(1);  
    connfd = Accept(listenfd, (struct sockaddr *) NULL, NULL); //Aceita a conexão, e inicia outro descritor de socket para esta conexão  
    if ((childpid = Fork()) == 0)  
    {  
        Close(listenfd);  
        sig_chld(connfd, servaddr);  
        exit(0);  
    }  
    Close(connfd);  
}  
return(0);  
}
```