

## **Exercício 4: Backlog e Processos Zumbis**

**Alunos:** Luiz Fernando Lima Leite e Mateus da  
Costa e Silva Rios Alves de Andrade

**RA:** 248405 e 230806

Instituto de Computação  
Universidade Estadual de Campinas

Campinas, 14 de Outubro de 2025.

# Sumário

1	Dinâmica de Filas de Conexão TCP . . . . .	2
2	Modificação do Servidor TCP . . . . .	3
2.1	Implementação sem tratamento de SIGCHLD . . . .	3
2.2	Implementação com tratamento de SIGCHLD . . . .	3
3	Testes Automatizados . . . . .	4
4	Testes com Sniffers . . . . .	5
5	Ambiente de Execução . . . . .	5

# 1 Dinâmica de Filas de Conexão TCP

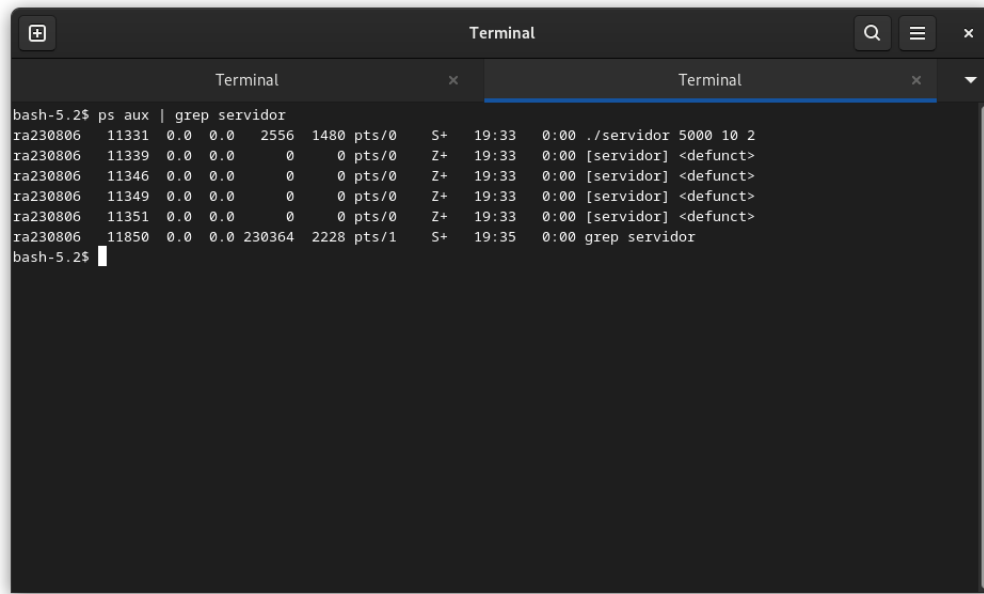
A SYN queue é uma fila de conexões TCP que ainda não completaram o Three-Way Handshake: assim que o kernel recebe uma requisição com a mensagem SYN ele adiciona a conexão na SYN queue. Já a Accept queue é a fila de conexões que concluíram o Three-Way Handshake e estão aguardando a chamada da função accept. Assim que o kernel recebe uma mensagem ACK do cliente ele remove a conexão da SYN queue e a adiciona na Accept queue, e, após uma chamada da função Accept, a conexão mais antiga da Accept queue é retirada da fila.

Essas filas existem para administrar dos recursos do servidor, permitindo que haja um controle da carga das tarefas de execução do Three-Way Handshake e processamento de requisições. Isso assegura o paralelismo do servidor concorrente e fornece resistência contra ataques de SYN flooding e DDoS.

O parâmetro backlog da função listen define o tamanho da Accept queue, enquanto que o parâmetro do kernel `/proc/sys/net/ipv4/tcp_max_syn_backlog` define o tamanho da SYN queue. O parâmetro backlog atua por socket, enquanto que o parâmetro do kernel é global e abrange todos os sockets do sistema. O kernel também possui um limite global de tamanho para a Accept queue, definido pelo parâmetro `/proc/sys/net/core/somaxconn`, e este limite não pode ser ultrapassado pelo parâmetro backlog da função listen.

## 2 Modificação do Servidor TCP

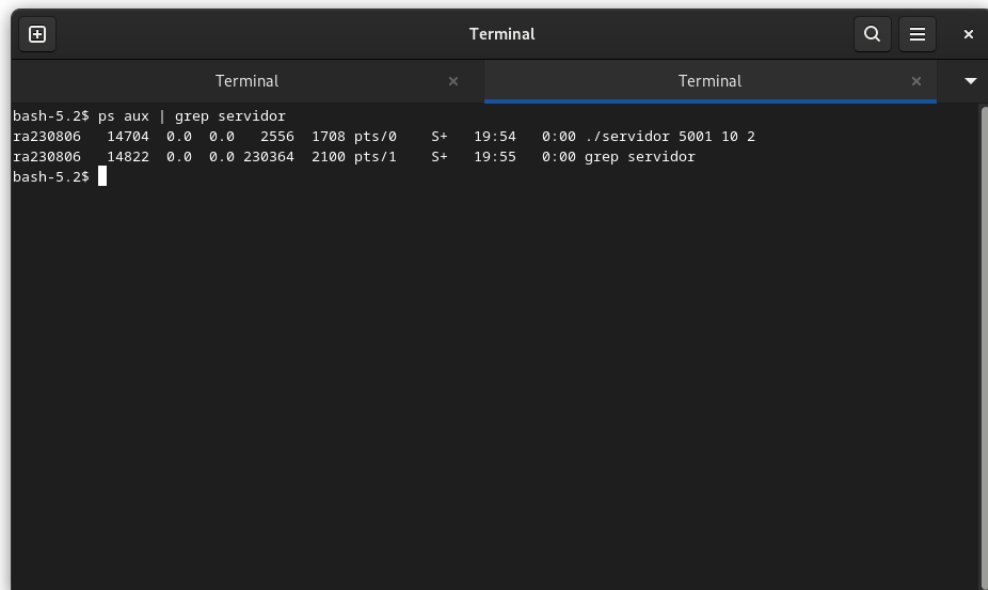
### 2.1 Implementação sem tratamento de SIGCHLD



```
bash-5.2$ ps aux | grep servidor
ra230806 11331 0.0 0.0 2556 1480 pts/0 S+ 19:33 0:00 ./servidor 5000 10 2
ra230806 11339 0.0 0.0 0 0 pts/0 Z+ 19:33 0:00 [servidor] <defunct>
ra230806 11346 0.0 0.0 0 0 pts/0 Z+ 19:33 0:00 [servidor] <defunct>
ra230806 11349 0.0 0.0 0 0 pts/0 Z+ 19:33 0:00 [servidor] <defunct>
ra230806 11351 0.0 0.0 0 0 pts/0 Z+ 19:33 0:00 [servidor] <defunct>
ra230806 11850 0.0 0.0 230364 2228 pts/1 S+ 19:35 0:00 grep servidor
bash-5.2$
```

### 2.2 Implementação com tratamento de SIGCHLD

O processo pai, ao receber o sinal SIGCHLD indicando que um processo filho terminou, chama a função `waitpid` para retirar esse processo do estado zumbi. Isso é feito em um loop, pois um outro processo pode terminar antes do handler retornar. A opção `WNOHANG` garante que, se nenhum processo tiver terminado, o servidor vai continuar sua execução ao invés de esperar que outro processo termine.



```
bash-5.2$ ps aux | grep servidor
ra230806 14704 0.0 0.0 2556 1708 pts/0 S+ 19:54 0:00 ./servidor 5001 10 2
ra230806 14822 0.0 0.0 230364 2100 pts/1 S+ 19:55 0:00 grep servidor
bash-5.2$
```

### 3 Testes Automatizados

Resultados dos testes variando o backlog (10 clientes simultâneos, delay de 1 segundo no servidor)

```
bash-5.2$ bash teste.sh 40400
127.0.0.1 40400
tcp_max_syn_backlog do sistema: 2048

Backlog | Conexões ESTABLISHED | Conexões Rejeitadas
-----|-----|-----
0       | 7                     | 3
1       | 10                    | 0
2       | 10                    | 0
3       | 10                    | 0
4       | 10                    | 0
5       | 10                    | 0
6       | 10                    | 0
7       | 10                    | 0
8       | 10                    | 0
9       | 10                    | 0
10      | 10                    | 0
```

Sem backlog, o servidor não foi rápido o bastante para processar todos os pedidos de conexão, resultando em três conexões rejeitadas inicialmente. Permitindo um backlog, isso não ocorreu e todas as conexões

foram feitas imediatamente. O tamanho máximo do backlog não afetou os testes, pois é ordens de magnitude maior do que os valores de backlog testados.

## **4 Testes com Sniffers**

Os clientes cujas conexões são rejeitadas só enviam mensagens com a flag SYN. Quando a fila está cheia, o servidor ignora novas conexões, forçando os clientes a esperar o timeout e enviar novas mensagens SYN até receber uma resposta.

Foi utilizado o comando "tcpdump -i lo tcp port 40400 and "tcp[tcpflags] & (tcp-syn) != 0" para inspecionar os testes. A saída está disponível no arquivo "tcpdump.out". Só observamos a primeira rodada de testes, visto que os resultados de uma se aplicam para todas.

## **5 Ambiente de Execução**

Foi utilizada a máquina "freitas"(143.106.16.232) para a execução dos testes.