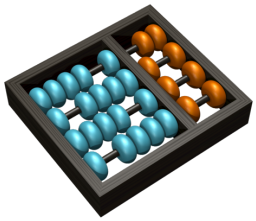

Atividade 4: Backlog e Processos Zumbis

Alunos: Elias Santos Martins e Gabriel Sanders Pereira Sobral

RA: 247057 e 247118

Instituto de Computação

Universidade Estadual de Campinas

Campinas, 19 de Agosto de 2025.

Sumário

1 Dinâmica das filas de conexão TCP no Kernel Linux

1.1 Fila de conexões incompletas (SYN queue) e fila de conexões completas (accept queue)

No Kernel Linux, temos duas filas de conexões:

- **Fila de conexões incompletas:** fila de conexões que foram socilitadas por clientes, mas cujo 3 way handshake ainda não está completo. Uma conexão é colocada nessa fila ao se receber o primeiro sinal `SYN` de um cliente
- **Fila de conexões completas:** fila de conexões cujo 3 way handshake foi completo. Uma conexão é movida da fila de conexões incompletas para esta fila ao se receber o primeiro `ACK` do cliente, que finaliza o 3 way handshake.

1.2 Parâmetro backlog da função listen()

O parâmetro backlog determina o tamanho máximo das filas de conexão, de modo que, quando a soma das duas filas ultrapssam o backlog, o servidor simplesmente ignora o SYN.

1.3 Parâmetro `/proc/sys/net/ipv4/tcp_max_syn_backlog`

A partir do Linux versão 2.2, o parâmetro *backlog* da função *listen* passou a definir somente o tamanho máximo da fila de conexões completas (https://linux.die.net/man/2/listen). O parâmetro */proc/sys/net/ipv4/tcp_max_syn_backlog*

pertence ao kernel e define o tamanho máximo da fila de conexões incompletas.

2 Servidor TCP concorrente com `fork()`

2.1 Backlog como argumento de linha de comando

Implementamos o backlog como o primeiro parâmetro da execução do servidor: .

```
servidor <backlog>
```

2.2 Configuração de tempo de espera (sleep)

Implementamos o backlog como o segundo parâmetro da execução do servidor: .

```
servidor <backlog> <tempo_sleep>
```

2.3 Execução sem tratador de sinais para SIGCHLD

Executamos o servidor e vários clientes, sem o tratamento de sinal. Podemos ver na figura 1 os clientes zumbis

2.4 Implementação do tratador de sinais para SIGCHLD

Implementamos o tratador de sinais para SIGCHLD no servidor, e fizemos o mesmo procedimento de executar vários clientes em sequência. Podemos ver agora na listagem do *ps* que não há processos zumbis.

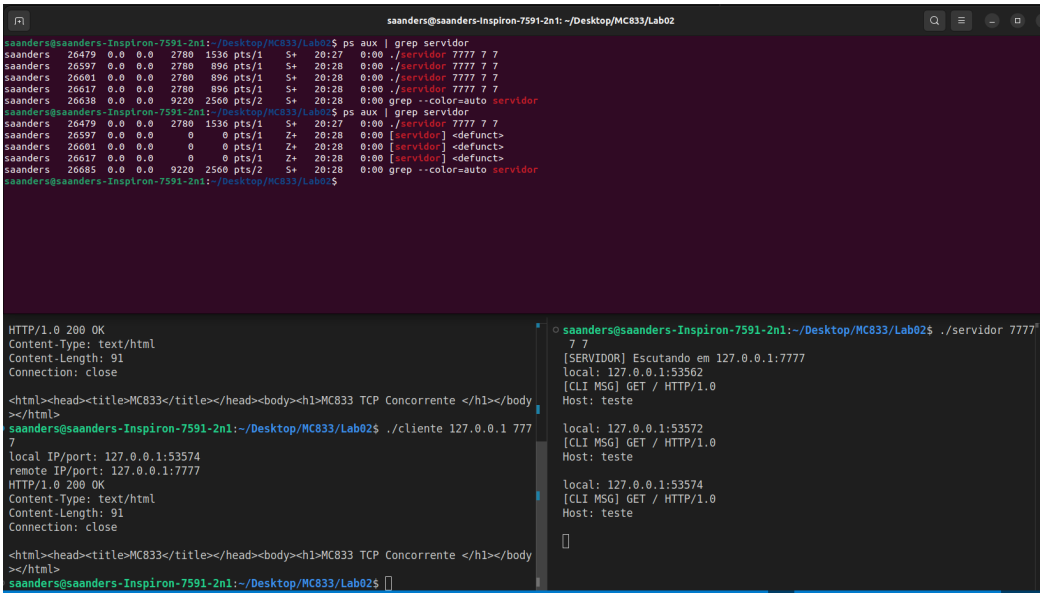

Figura 1: Processos zumbis gerados por clientes encerrados

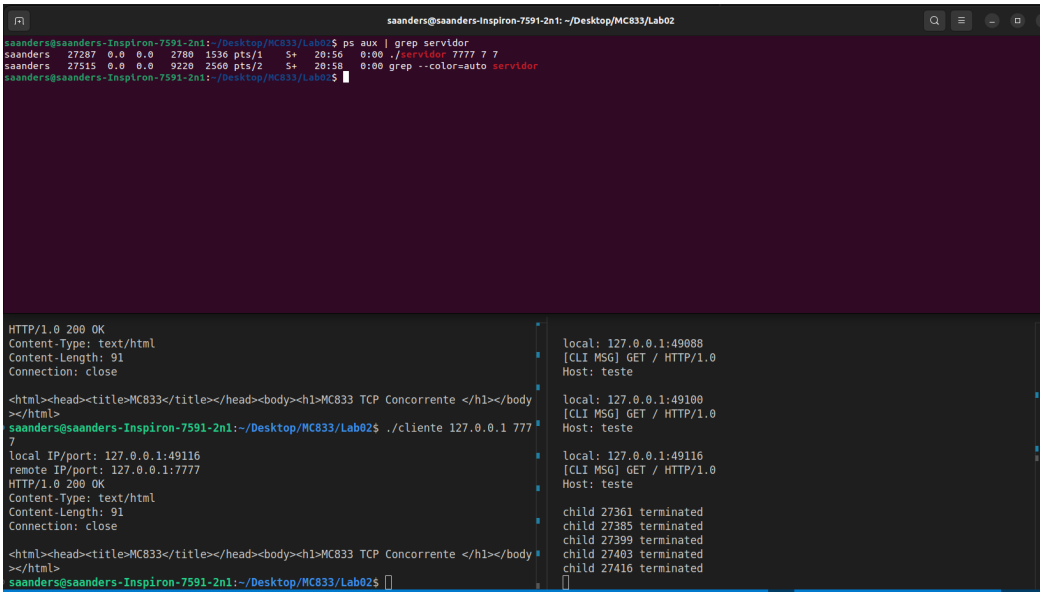

Figura 2: Ausência de processos zumbis após tratamento do SIGCHLD pelo servidor

2.5 Explicação sobre processos zumbis

Um processo zumbi é um processo que já foi encerrado, mas cujo pai ainda não recebeu o sinal de retorno. Portanto, é mantido uma entrada na tabela de processos para esse processo, que fica marcado com o estado Z (zumbi). O processo só será removido dessa tabela quando o processo pai correspondente chamar o *wait()*. Se o pai não está configurado para receber a notificação de encerramento do processo filho, o processo filho permanecerá na tabela, no estado de zumbi.

3 Automatização de testes de carga com script

3.1 Execução com diferentes valores de backlog

Para lancar vários clientes simultâneos em diferentes valores de backlog, implementamos um script bash, que executa um servidor para cada valor de backlog indo de 0 até 10. Para cada servidor, rodamos o comando `seq 10 | xargs -P10 -I ./cliente 127.0.0.1 7777` para executar 10 clientes simultaneamente. Colocamos um sleep de 5 segundos no cliente antes de ele fechar a conexão, para ser possível capturar a conexão no estado de `ESTABILISHED`.

3.2 Tabela de resultados

Tabela 1: Resultados da variação do backlog no servidor TCP

Backlog	Conexões imediatas	Conexões rejeitadas
0	10	0
1	3	0
2	4	0
3	5	0
4	6	0
5	7	0
6	8	0
7	9	0
8	10	0
9	10	0
10	10	0

Podemos ver que o valor de conexões imediatas não bate exatamente com o valor do backlog, mas é proporcional (sempre 2 a mais).

3.3 Comparação com `/proc/sys/net/ipv4/tcp_max_syn_backlog`

O parâmetro `backlog` cuida do tamanho da fila de conexões completas, enquanto o `tcp_max_syn_backlog` da fila de conexões incompletas. No nosso experimento alteramos apenas o tamanho da fila de conexões completas, e a fila de conexões incompletas ficou com seu tamanho padrão

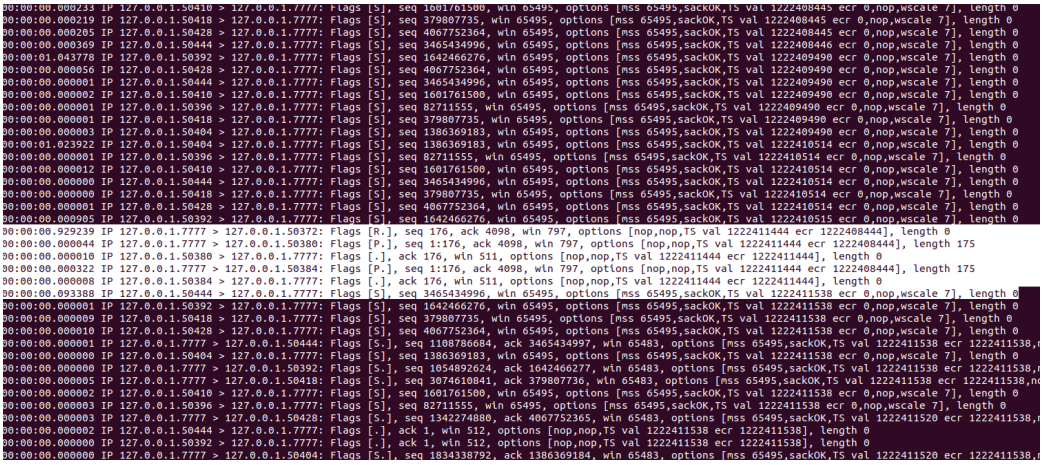

Figura 3: Sniffer durante os testes: *sudo tcpdump -i lo -n -ttt 'tcp and port 7777'*

4 Análise de pacotes com sniffer

4.1 Flags TCP observadas em clientes sem conexão

Quando um cliente não consegue se conectar, então aparece a flag [R.] e [S.], a conexão foi recusada e que uma retransmissão do SYNC está sendo feita, como demonstrado na figura 3. Quando a fila está cheia, aparecem apenas SYNC retransmitidos pois o cliente continua tentando estabelecer uma conexão, e faz isso até que o servidor libere espaço em sua fila.