

| <b>Disciplina: Sistemas Operacionais    Profa. Alana Oliveira</b> |                      |
|---|----------------------|
| Nome do aluno:  | Israel Barbosa Silva |
| Nome do aluno:  | Israel Barbosa Silva |
| Nome do aluno:  | Israel Barbosa Silva |
| Sistema Operacional escolhido:                                    | Ubuntu 22.04         |
| Data:   | 06/07/2025           |

## 1 Gerência de Processos

### Como o sistema cria e gerencia processos?

O Ubuntu adota um modelo clássico de criação de processos usando as chamadas de sistema `fork()` e `exec()`. A função `fork()` gera uma cópia quase exata do processo pai, resultando em um processo filho com um novo PID, enquanto `exec()` substitui o espaço de memória do filho com um novo programa, mantendo o mesmo PID.

Por exemplo, quando você digita `ls` no terminal, o shell (como o `bash`) utiliza `fork()` para criar um processo filho que, em seguida, executa `exec()` para carregar e executar o comando `ls`, com o pai aguardando a finalização do filho.

Internamente, o Kernel gerencia cada processo através da estrutura `task_struct`, que contém informações essenciais como estado, memória e arquivos abertos. O escalonador, como o Completely Fair Scheduler (CFS), distribui o tempo de CPU entre os processos, garantindo uma execução equilibrada e eficiente.

### Há suporte a threads, multitarefa ou paralelismo?

O Ubuntu oferece suporte robusto para multitarefa preemptiva, permitindo que o kernel interrompa processos em execução para evitar que um único processo monopolize a CPU. Isso assegura que múltiplos processos possam ser executados "ao mesmo tempo", mantendo o sistema responsivo.

Além disso, o sistema possui um forte suporte a threads, que são tratadas como processos leves (LWPs) e compartilham recursos como memória e descritores de arquivo. A biblioteca padrão para criação de threads é a `pthread`, facilitando a comunicação rápida e eficiente entre elas.

O hardware moderno com múltiplos núcleos possibilita o paralelismo verdadeiro, já que o kernel pode agendar diferentes processos ou threads para serem executados simultaneamente, ampliando o poder de processamento e a capacidade de resposta do sistema.

### Como o usuário ou o administrador pode visualizar, controlar ou encerrar processos?

#### **ps**

- **Descrição:** Exibe uma lista de processos em execução no sistema, permitindo visualizar informações como PID, uso de CPU e memória.
- **Comando:** `ps aux`

- **Informações Exibidas:**
  - USER: Usuário que executa o processo
  - PID: Identificador do processo
  - %CPU: Percentual de uso da CPU
  - %MEM: Percentual de uso da memória
  - COMMAND: Comando que iniciou o processo
  - VSZ: Tamanho virtual do processo
  - RSS: Tamanho residente na memória física
  - STAT: Estado do processo
  - START: Hora de início do processo
  - TIME: Tempo total de CPU usado pelo processo
  - TTY: Terminal associado ao processo

```
israelbsi@DESKTOP-QEPT4RR:~$ ps aux
USER      PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root         1  0.0  0.0 22072 13244 ?        Ss   12:25   0:00 /sbin/init
root         2  0.0  0.0  2616  1444 ?        Sl   12:25   0:00 /init
root         7  0.0  0.0  2616   132 ?        Sl   12:25   0:00 plan9 --control-socket 6 --log-level 4 --server-fd 7
root        49  0.0  0.1 66836 16584 ?        S<s  12:25   0:00 /usr/lib/systemd/systemd-journald
root        75  0.0  0.0 24124  6280 ?        Ss   12:25   0:00 /usr/lib/systemd/systemd-udev
systemd+   157  0.0  0.0 21452 11800 ?        Ss   12:25   0:00 /usr/lib/systemd/systemd-resolved
systemd+   158  0.0  0.0  91020 6560 ?        Ssl  12:25   0:00 /usr/lib/systemd/systemd-timesyncd
root       164  0.0  0.0  4236  2652 ?        Ss   12:25   0:00 /usr/sbin/cron -f -P
message+   165  0.0  0.0  9784  5408 ?        Ss   12:25   0:00 @dbus-daemon --system --address=systemd: --nofork --n
root       173  0.0  0.0 17976  8248 ?        Ss   12:25   0:00 /usr/lib/systemd/systemd-logind
```

top

- **Descrição:** Exibe em tempo real os processos ativos, uso de CPU e memória, permitindo identificar quais processos estão consumindo mais recursos.
- **Comando:** `top`
- **Informações Exibidas:**
  - Lista de processos atualizada dinamicamente
  - Uso detalhado de CPU
  - Uso de memória
  - Tempo de execução dos processos

```
israelbsi@DESKTOP-QEPT4RR:~$ top
top - 12:38:27 up 34 min, 1 user, load average: 0.00, 0.00, 0.00
Tasks: 23 total, 1 running, 22 sleeping, 0 stopped, 0 zombie
%Cpu(s): 0.0 us, 0.0 sy, 0.0 ni,100.0 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st
MiB Mem : 15957.2 total, 15321.3 free, 647.8 used, 249.0 buff/cache
MiB Swap: 4096.0 total, 4096.0 free, 0.0 used. 15309.4 avail Mem

  PID USER      PR  NI  VIRT  RES  SHR S  %CPU  %MEM    TIME+  COMMAND
    1 root        20   0  21668 12892 9520 S   0.0   0.1   0:00.32 systemd
    2 root        20   0   2616  1444 1320 S   0.0   0.0   0:00.00 init-systemd(Ub
    7 root        20   0   2616   132  132 S   0.0   0.0   0:00.00 init
   49 root       19  -1 66836 15988 14816 S   0.0   0.1   0:00.18 systemd-journal
   75 root        20   0 23992  6192 4936 S   0.0   0.0   0:00.08 systemd-udev
  157 systemd+    20   0 21452 11800 9600 S   0.0   0.1   0:00.06 systemd-resolve
  158 systemd+    20   0 91020 6560 5512 S   0.0   0.0   0:00.06 systemd-timesyn
```

htop

- **Descrição:** Uma versão aprimorada do top, com uma interface mais amigável e recursos adicionais.
- **Comando:** `htop`
- **Recursos Adicionais:**
  - Permite ordenar processos por diferentes critérios (CPU, memória, etc.).
  - Possibilita o encerramento de processos diretamente pela interface.

```

0[          0.0%] 3[          0.0%] 6[          0.0%] 9[          0.0%]
1[          0.0%] 4[          0.0%] 7[          0.0%] 10[         0.0%]
2[          0.0%] 5[          0.0%] 8[          0.0%] 11[         0.0%]
Mem[|||||]      400M/15.6G Tasks: 25, 22 thr, 0 kthr; 1 running
Swp[          0K/4.00G] Load average: 0.07 0.02 0.00
                          Uptime: 00:37:09


Main  I/O
PID  USER    PRI  NI  VIRT  RES  SHR  S  CPU% MEM%  TIME+  Command
1    root     20    0  21668 12896 9520 S   0.0  0.1  0:00.34 /sbin/init
2    root     20    0  2616  1444  1320 S   0.0  0.0  0:00.00 /init
7    root     20    0  2616  132  132 S   0.0  0.0  0:00.00 plan9 --control-socket 6 --log-level 4 --server-fd 7
8    root     20    0  2616  132  132 S   0.0  0.0  0:00.00 plan9 --control-socket 6 --log-level 4 --server-fd 7
9    root     20    0  2616  1444  1320 S   0.0  0.0  0:00.00 /init
49   root     19   -1 66836 16000 14828 S   0.0  0.1  0:00.18 /usr/lib/systemd/systemd-journald
75   root     20    0 23992  6192  4936 S   0.0  0.0  0:00.08 /usr/lib/systemd/systemd-udevd
157  systemd-re 20    0 21452 11800  9600 S   0.0  0.1  0:00.07 /usr/lib/systemd/systemd-resolved
158  systemd-ti 20    0 91020  6560  5712 S   0.0  0.0  0:00.06 /usr/lib/systemd/systemd-timesyncd
162  systemd-ti 20    0 91020  6560  5712 S   0.0  0.0  0:00.00 /usr/lib/systemd/systemd-timesyncd
164  root     20    0  4236  2652  2420 S   0.0  0.0  0:00.00 /usr/sbin/cron -f -P
165  messagebus 20    0  9660  5124  4504 S   0.0  0.0  0:00.07 @dbus-daemon --system --address=systemd: --nofork --n
173  root     20    0 17976  8248  7232 S   0.0  0.1  0:00.06 /usr/lib/systemd/systemd-logind
176  root     20    0 1714M 16004  9392 S   0.0  0.1  0:00.05 /usr/libexec/wsl-pro-service -vv
182  root     20    0  3160  1048  960 S   0.0  0.0  0:00.00 /sbin/agetty -o -p -- \u --noclear --keep-baud - 1152
195  root     20    0  3116  1016  932 S   0.0  0.0  0:00.00 /sbin/agetty -o -p -- \u --noclear - linux
202  syslog    20    0  217M  5284  4432 S   0.0  0.0  0:00.02 /usr/sbin/rsyslogd -n -iNONE
211  root     20    0 104M  22500 13164 S   0.0  0.1  0:00.05 /usr/bin/python3 /usr/share/unattended-upgrades/unatt
215  root     20    0 1714M 16004  9392 S   0.0  0.1  0:00.00 /usr/libexec/wsl-pro-service -vv
F1Help F2Setup F3Search F4Filter F5Tree F6SortBy F7Nice F8Nice F9Kill F10Quit

```

## gnome-system-monitor

- **Descrição:** Uma ferramenta gráfica para monitoramento de sistema que exibe informações sobre processos, uso de CPU, memória e rede.
- **Comando:** `gnome-system-monitor`

| Processes Resources File Systems Q ≡ - □ × |           |       |      |          |                |           |
|--|-----------|-------|------|----------|----------------|-----------|
| Process Name                               | User      | % CPU | ID   | Memory   | Disk read tota | Disk writ |
| (sd-pam)                                   | israelbsi | 0.00  | 383  | 1.8 MB   | N/A            |           |
| at-spi-bus-launcher                        | israelbsi | 0.00  | 1609 | 2.7 MB   | 32.8 kB        |           |
| at-spi2-registryd                          | israelbsi | 0.00  | 1638 | 692.2 kB | 352.3 kB       |           |
| bash                                       | israelbsi | 0.00  | 293  | 1.8 MB   | 21.1 MB        |           |
| bash                                       | israelbsi | 0.00  | 394  | 1.7 MB   | 8.2 kB         |           |
| dbus-daemon                                | israelbsi | 0.00  | 1598 | 405.5 kB | 61.4 kB        |           |
| dbus-daemon                                | israelbsi | 0.00  | 1616 | 401.4 kB | 20.5 kB        |           |
| dconf-service                              | israelbsi | 0.00  | 1623 | 466.9 kB | 77.8 kB        | 12        |
| gnome-system-monitor                       | israelbsi | 0.03  | 1579 | 79.8 MB  | 112.1 MB       |           |
| gvfs-udisks2-volume-monitor                | israelbsi | 0.00  | 1617 | 1.3 MB   | N/A            |           |
| gvfsd                                      | israelbsi | 0.00  | 1600 | 729.1 kB | N/A            |           |
| systemd                                    | israelbsi | 0.00  | 381  | 2.1 MB   | N/A            |           |

End Process


### kill, killall, pkill

- **Descrição:** Comandos para enviar sinais a processos, permitindo encerrá-los ou alterar seu comportamento.
- **Comando:** `kill <PID>`, `killall <nome_do_processo>`, `pkill <nome_do_processo>`

```
israelbsi@DESKTOP-QEPT4RR x + v
israelbsi@DESKTOP-QEPT4RR:~$ killall
Usage: killall [OPTION]... [--] NAME...
    killall -l, --list
    killall -V, --version

-e,--exact          require exact match for very long names
-I,--ignore-case    case insensitive process name match
-g,--process-group  kill process group instead of process
-y,--younger-than   kill processes younger than TIME
-o,--older-than     kill processes older than TIME
-i,--interactive    ask for confirmation before killing
-l,--list           list all known signal names
-q,--quiet          don't print complaints
-r,--regex          interpret NAME as an extended regular expression
-s,--signal SIGNAL  send this signal instead of SIGTERM
-u,--user USER      kill only process(es) running as USER
-v,--verbose        report if the signal was successfully sent
-V,--version        display version information
-w,--wait           wait for processes to die
-n,--ns PID         match processes that belong to the same namespaces
                    as PID
-Z,--context REGEXP kill only process(es) having context
                    (must precede other arguments)

israelbsi@DESKTOP-QEPT4RR:~$ kill
kill: usage: kill [-s sigspec | -n signum | -sigspec] pid | jobspec ... or kill -l [sigspec]
israelbsi@DESKTOP-QEPT4RR:~$ |
```

## 2 Gerência de Memória

O sistema usa memória virtual?

Sim, a memória virtual simula um espaço contíguo e extenso para cada processo, independentemente da quantidade física instalada. Isso permite que cada aplicação opere como se tivesse acesso a uma grande capacidade de memória, garantindo vantagens como o isolamento entre processos e a otimização do uso da RAM.

- **Isolamento de Processos:** Cada processo possui seu próprio espaço de endereçamento virtual, impedindo acesso não autorizado entre processos e aumentando a segurança e estabilidade do sistema.
- **Eficiência no Uso da RAM:** O sistema carrega na memória apenas as partes do programa que estão em uso, utilizando a área de swap para armazenar dados não ativos, otimizando o consumo de memória.

Há paginação ou segmentação?

- No Linux moderno, a segmentação é mínima devido ao modelo flat, sendo a paginação o principal mecanismo de gerenciamento de memória. A memória é dividida em páginas de tamanho fixo (geralmente 4 KiB), e o sistema utiliza tabelas de páginas para mapear os endereços virtuais para os físicos. O carregamento sob demanda permite que as páginas sejam trazidas do disco para a RAM apenas quando necessário, otimizando o desempenho e o uso dos recursos.

Como ocorre o gerenciamento entre processos ativos e memória disponível?

- O Kernel Linux gerencia a memória atribuída aos processos por meio de mecanismos que alocam, liberam e realocam recursos conforme a demanda. Inicialmente, ele reserva as estruturas necessárias, como tabelas de páginas, e, à medida que os processos requisitam mais memória, novas páginas são mapeadas. Quando a memória RAM esgota, o sistema utiliza o swap, transferindo páginas inativas para o disco, embora com uma queda de desempenho devido à maior latência de acesso.
- Além disso, o Linux agiliza o acesso a dados com o cache de página, que armazena arquivos lidos para acelerar futuras operações, e incorpora o OOM Killer, um mecanismo de emergência que encerra processos para liberar memória quando os recursos estão completamente esgotados.

Há ferramentas para monitoramento (ex: top, free, etc.)?

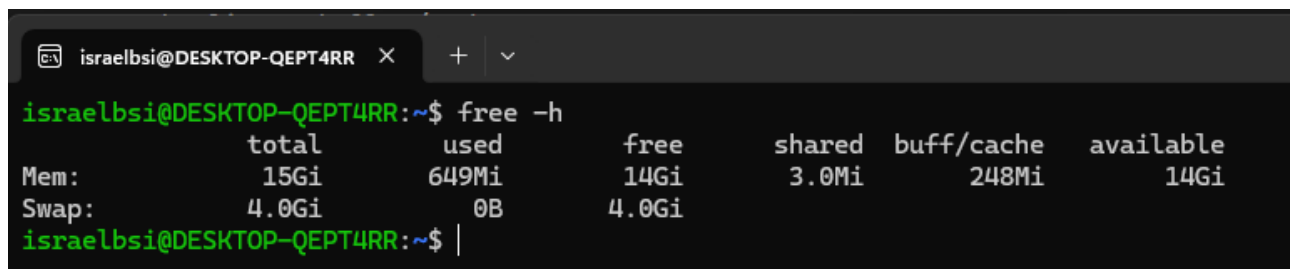
Sim, o Ubuntu 22.04 oferece várias ferramentas para monitoramento de memória e processos:

### free

- **Descrição:** Exibe uma visão geral do uso de memória no sistema.
- **Comando:** `free -h`

### Informações Exibidas

- **total:** Total de RAM física.
- **used:** Memória usada por processos.
- **free:** Memória efetivamente não utilizada.
- **shared:** Memória compartilhada entre processos.
- **buff/cache:** Memória usada para cache de disco e buffers. Essa memória pode ser liberada se necessário.
- **available:** Estimativa da memória disponível para novos aplicativos sem recorrer ao swap.
- **Swap:** Exibe o total, o usado e o livre do espaço de troca.



```
israelbsi@DESKTOP-QEPT4RR:~$ free -h
              total        used        free       shared    buff/cache   available
Mem:           15Gi        649Mi        14Gi         3.0Mi         248Mi         14Gi
Swap:          4.0Gi           0B         4.0Gi
```

### vmstat

- **Descrição:** Exibe estatísticas sobre processos, memória, paginação, blocos de E/S e CPU, permitindo uma análise mais detalhada do desempenho do sistema.
- **Comando:** `vmstat`

```
israelbsi@DESKTOP-QEPT4RR:~$ vmstat
procs -----memory----- --swap-- ----io---- -system-- -----cpu-----
 r  b   swpd   free   buff   cache   si   so    bi   bo    in   cs   us   sy   id   wa   st   gu
  0   0       0 15559180 16036 345708    0    0   194   40   35    0    0    0  100    0    0    0
israelbsi@DESKTOP-QEPT4RR:~$
```

### top, htop, gnome-system-monitor

- Estas ferramentas já foram discutidas na seção de Gerência de Processos, mas também fornecem informações sobre o uso de memória pelos processos em execução.

## 3 Gerência de Arquivos

Qual é o sistema de arquivos utilizado (ex: NTFS, ext4, Btrfs...)?

Como os arquivos são organizados?

Há suporte a permissões por usuário/grupo?

Quais operações básicas são disponíveis?

## 4 Proteção e Segurança

Como o sistema trata a autenticação de usuários?

Há mecanismos de criptografia ou controle de acesso?

Existe separação clara entre usuários comuns e administradores?

Há antivírus, firewall, SELinux, AppArmor, UAC etc.?

## Comparativo com o que foi estudado

- 

## Fontes

### Documentação

- [Documentação do Kernel Linux Sobre Gerenciamento de Memória](#)
- [Documentação do Kernel Linux sobre Scheduler](#)
- Manual do Comando `man free`: [free\(1\) — Linux manual pages](#)
- Manual do Comando `man top`: [top\(1\) — Linux manual pages](#)
- Manual do Comando `man htop`: [htop\(1\) — Linux manual pages](#)
- Manual do Comando `man vmstat`: [vmstat\(8\) — Linux manual pages](#)

- Manual do Comando `man` `gnome-system-monitor`: [Monitor do Sistema](#)
- Manual do Comando `man` `2` `fork`: [fork\(2\) — Linux manual pages](#)
- Manual do Comando `man` `3` `exec`: [exec\(3\) — Linux manual pages](#)
- [Artigo sobre o OOM Killer](#)
- [Artigo sobre a implementação de Threads no Linux](#)

## Livros

- Love, Robert. Linux Kernel Development, 3rd Edition. Addison-Wesley Professional, 2010.
- Bovet, Daniel P., e Cesati, Marco. Understanding the Linux Kernel, 3rd Edition. O'Reilly Media, 2005