

Gerenciamento de processos e tarefas

Todo processo é um **programa** em execução, e cada processo tem um **conjunto de estados**.

O SO é responsável pelo gerenciamento de todas as informações necessárias para a execução de qualquer **programa**.

Tipos de Sistemas:

MonoTarefa

Permite a execução de um único programa/tarefa.

Somente um programa pode ser carregado para a memória

MultiTarefas

Permite a execução de vários programas/tarefas.

Permite que vários programas sejam carregados para a memória.

Metodologia de Execução:

Cooperativa

O sistema operacional **nunca inicia** a alternância de contexto do programa em execução para outro programa.

Preemptiva

O sistema operacional **pode iniciar** uma mudança de contexto do programa em execução para outro programa.

Multithreading

Executa mais do que uma tarefa de um único programa.

Tarefas distintas são criadas nos programas em execução.

Processos são entidades independentes.

Cada um possui permissões de acesso e atributos (características).

No Linux, um processo é **uma instância de um software em execução**, ou seja, um programa que está sendo usado. Em outros sistemas, os processos também ganham o nome de tarefas (tasks).

No Windows, temos o Gerenciador de Tarefas, que auxilia por exemplo a fechar um aplicativo que travou ou está em estado indesejável.

Em Linux, um sistema multitarefa, também possui recursos semelhantes.

TOP (top - display Linux processes)

Execute no Terminal:

\$ **top**

O comando top é a maneira mais comum de verificar o uso de processos do sistema e quais deles estão consumindo mais memória ou processamento.

Estados dos processos ou tasks:

1. Ready (scheduler tasks kernel)
2. **R**unning
3. **S**leeping (not use)
4. **S**topped
5. **Z**ombie (PPID is kill)
6. Waiting (start and stop thread)
7. Dead (comand kill)

```
urubu100@DESKTOP-00I5LD3: ~  
top - 15:09:31 up 31 min, 0 users, load average: 0.52, 0.58, 0.59  
Tasks: 5 total, 1 running, 4 sleeping, 0 stopped, 0 zombie  
%Cpu(s): 54.9 us, 4.2 sy, 0.0 ni, 40.8 id, 0.0 wa, 0.1 hi, 0.0 si, 0.0 st  
KiB Mem : 8321388 total, 2735972 free, 5348940 used, 236476 buff/cache  
KiB Swap: 10215804 total, 9918844 free, 296960 used. 2831592 avail Mem  


| PID | USER     | PR | NI | VIRT  | RES  | SHR  | S | %CPU | %MEM | TIME+   | COMMAND |
|-----|----------|----|----|-------|------|------|---|------|------|---------|---------|
| 1   | root     | 20 | 0  | 8324  | 152  | 128  | S | 0.0  | 0.0  | 0:00.15 | init    |
| 3   | root     | 20 | 0  | 8328  | 152  | 116  | S | 0.0  | 0.0  | 0:00.01 | init    |
| 4   | urubu100 | 20 | 0  | 15220 | 3740 | 3632 | S | 0.0  | 0.0  | 0:00.22 | bash    |
| 44  | urubu100 | 20 | 0  | 15160 | 3700 | 3608 | S | 0.0  | 0.0  | 0:00.07 | bash    |
| 75  | urubu100 | 20 | 0  | 15900 | 1940 | 1408 | R | 0.0  | 0.0  | 0:00.03 | top     |

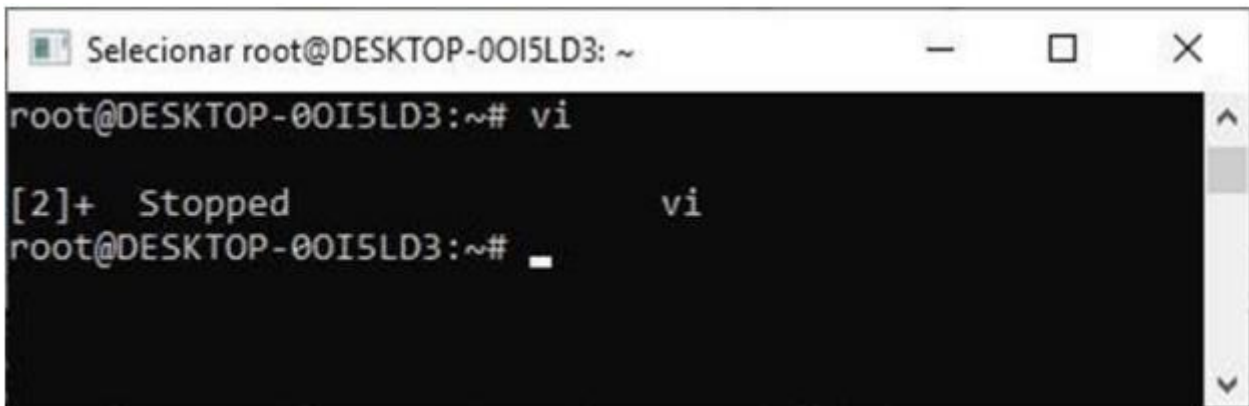

```

Vamos iniciar um processo e parar.

Execute como root

vi

E depois dê Ctrl + z (comando que suspende o processo, diferente do Ctrl + c que encerra o processo)



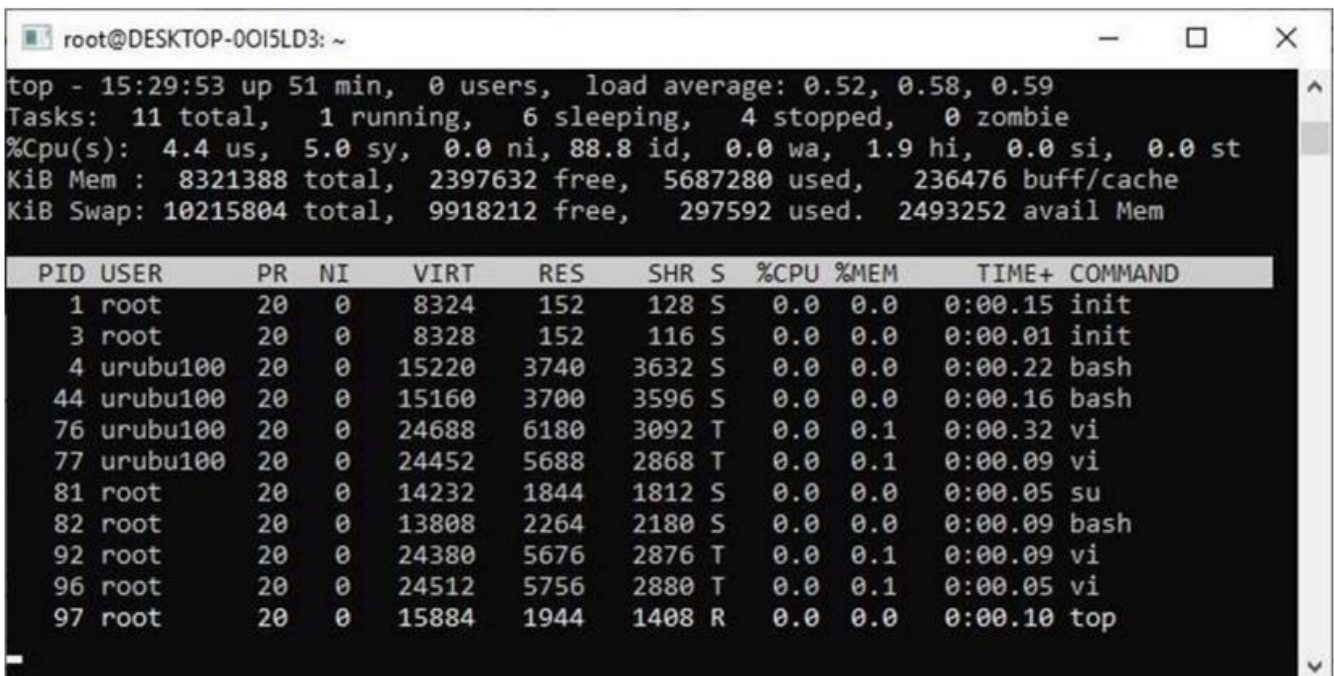
```
Selecionar root@DESKTOP-00I5LD3: ~
root@DESKTOP-00I5LD3:~# vi

[2]+  Stopped                  vi
root@DESKTOP-00I5LD3:~# _
```

Veja que o processo do editor de vi foi iniciado e depois parado.

Novamente execute o comando

top



```
root@DESKTOP-00I5LD3: ~
top - 15:29:53 up 51 min, 0 users, load average: 0.52, 0.58, 0.59
Tasks: 11 total, 1 running, 6 sleeping, 4 stopped, 0 zombie
%Cpu(s): 4.4 us, 5.0 sy, 0.0 ni, 88.8 id, 0.0 wa, 1.9 hi, 0.0 si, 0.0 st
KiB Mem : 8321388 total, 2397632 free, 5687280 used, 236476 buff/cache
KiB Swap: 10215804 total, 9918212 free, 297592 used. 2493252 avail Mem

  PID USER      PR  NI   VIRT   RES   SHR  S  %CPU  %MEM    TIME+  COMMAND
    1 root        20   0    8324    152    128  S   0.0   0.0   0:00.15  init
    3 root        20   0    8328    152    116  S   0.0   0.0   0:00.01  init
    4 urubu100    20   0   15220   3740   3632  S   0.0   0.0   0:00.22  bash
   44 urubu100    20   0   15160   3700   3596  S   0.0   0.0   0:00.16  bash
   76 urubu100    20   0   24688   6180   3092  T   0.0   0.1   0:00.32  vi
   77 urubu100    20   0   24452   5688   2868  T   0.0   0.1   0:00.09  vi
   81 root        20   0   14232   1844   1812  S   0.0   0.0   0:00.05  su
   82 root        20   0   13808   2264   2180  S   0.0   0.0   0:00.09  bash
   92 root        20   0   24380   5676   2876  T   0.0   0.1   0:00.09  vi
   96 root        20   0   24512   5756   2880  T   0.0   0.1   0:00.05  vi
   97 root        20   0   15884   1944   1408  R   0.0   0.0   0:00.10  top
```

Observe os estados dos processos:

1 processo rodando, **6** em sleeping, **4** stopped e **0** em zombie

Temos o total de memória RAM usada e livre

Temos o total de memória ou área de swap ocupada e livre.

PID é o número do processo.

User: a qual usuário está associado aquele **PID**, é o proprietário daquele processo.

PR: Prioridade **R**eal sob a perspectiva do kernel. **PR** pode variar entre -20 a +20, sendo que - 20 tem uma prioridade mais alta e +20 prioridades mais baixa.

Quando o valor é igual a zero é neutro/padrão.

Em linux temos 140 prioridades, que podem ser classificadas:

Alto: -20 -100

Padrão: 0 120

Inferior: +19 +139

NI - nice prioridade do comando

Execute o comando

nice -n -20 vi

```
root@DESKTOP-00I5LD3: ~
top - 16:32:15 up 1:53, 0 users, load average: 0.52, 0.58, 0.59
Tasks: 12 total, 1 running, 6 sleeping, 5 stopped, 0 zombie
%Cpu(s): 4.1 us, 4.6 sy, 0.0 ni, 89.7 id, 0.0 wa, 1.5 hi, 0.0 si, 0.0 st
KiB Mem : 8321388 total, 3202992 free, 4881920 used, 236476 buff/cache
KiB Swap: 10215804 total, 9771644 free, 444160 used. 3298612 avail Mem

  PID USER      PR  NI  VIRT  RES  SHR  S  %CPU  %MEM   TIME+ COMMAND
    1 root        20   0   8324   152   128  S   0.0   0.0   0:00.15 init
    3 root        20   0   8328   152   116  S   0.0   0.0   0:00.01 init
    4 urubu100    20   0  15220  3740  3632  S   0.0   0.0   0:00.22 bash
   44 urubu100    20   0  15160  3700  3596  S   0.0   0.0   0:00.16 bash
   76 urubu100    20   0  24688  6148  3092  T   0.0   0.1   0:00.32 vi
   77 urubu100    20   0  24452  5652  2868  T   0.0   0.1   0:00.09 vi
   81 root        20   0  14232  1844  1812  S   0.0   0.0   0:00.05 su
   82 root        20   0  13808  2272  2184  S   0.0   0.0   0:00.25 bash
   92 root        20   0  24528  5760  2888  T   0.0   0.1   0:00.10 vi
   96 root        20   0  24512  5708  2888  T   0.0   0.1   0:00.05 vi
  117 root        40  20  24356  5656  2868  T   0.0   0.1   0:00.06 vi
  119 root        20   0  15888  1932  1408  R   0.0   0.0   0:00.30 top
```


Execute o comando:

nice -n 20 vi

```
root@DESKTOP-00I5LD3: ~  
top - 16:37:10 up 1:58, 0 users, load average: 0.52, 0.58, 0.59  
Tasks: 14 total, 1 running, 6 sleeping, 7 stopped, 0 zombie  
%Cpu(s): 3.6 us, 3.2 sy, 0.0 ni, 92.6 id, 0.0 wa, 0.7 hi, 0.0 si, 0.0 st  
KiB Mem : 8321388 total, 3236476 free, 4848436 used, 236476 buff/cache  
KiB Swap: 10215804 total, 9742980 free, 472824 used. 3332096 avail Mem  
  
  PID USER      PR  NI   VIRT   RES   SHR  S  %CPU  %MEM    TIME+  COMMAND  
  121 root       20   0   15912   1976   1408 R   0.3   0.0   0:00.07 top  
    1 root       20   0    8324    152    128 S   0.0   0.0   0:00.15 init  
    3 root       20   0    8328    152    116 S   0.0   0.0   0:00.01 init  
    4 urubu100   20   0   15220   3740   3632 S   0.0   0.0   0:00.22 bash  
   44 urubu100   20   0   15160   3700   3596 S   0.0   0.0   0:00.16 bash  
   76 urubu100   20   0   24688   6148   3092 T   0.0   0.1   0:00.32 vi  
   77 urubu100   20   0   24452   5652   2868 T   0.0   0.1   0:00.09 vi  
   81 root       20   0   14232   1844   1812 S   0.0   0.0   0:00.05 su  
   82 root       20   0   13808   2272   2188 S   0.0   0.0   0:00.29 bash  
   92 root       20   0   24528   5760   2888 T   0.0   0.1   0:00.10 vi  
   96 root       20   0   24512   5708   2888 T   0.0   0.1   0:00.05 vi  
  117 root       40  20   24356   5656   2868 T   0.0   0.1   0:00.06 vi  
  119 root       20   0   15888   1932   1408 T   0.0   0.0   0:00.32 top  
  120 root        1 42+   24512   5744   2868 T   0.0   0.1   0:00.08 vi
```

O **VIRT** é a quantidade total de memória utilizada em cada processo, incluído bibliotecas, chamadas e cálculos.

O **RES** é a memória residente, sem considerar as trocas.

O **SHR** é a memória compartilhada entre processos.

Em **S**, temos o status do processo: **R** reflete processo rodando, **S** reflete o processo sleeping e **T** reflete o processo stopped.

Agora entre no editor **vi** novamente e depois **pare este processo** e:

Execute o comando:

jobs

```
root@DESKTOP-00I5LD3: ~  
root@DESKTOP-00I5LD3:~# jobs  
[1]+  Stopped                  vi  
[2]-  Stopped                  vi  
root@DESKTOP-00I5LD3:~#
```

jobs -l (exibe o nome e o número de cada processo)

```
root@DESKTOP-00I5LD3: ~  
root@DESKTOP-00I5LD3:~# jobs -l  
[1]+  92 Stopped                  vi  
[2]-  96 Stopped                  vi  
root@DESKTOP-00I5LD3:~#
```

jobs -s (exibe o nome de cada processo)

```
root@DESKTOP-00I5LD3: ~  
root@DESKTOP-00I5LD3:~# jobs -s  
[1]+  Stopped                    vi  
[2]-  Stopped                    vi  
root@DESKTOP-00I5LD3:~#
```

jobs -p (exibe o número de cada processo)

```
root@DESKTOP-00I5LD3: ~  
root@DESKTOP-00I5LD3:~# jobs -p  
92  
96  
root@DESKTOP-00I5LD3:~#
```

Vamos agora recordar sobre o comando de informações dos processos:

ps

Exibe informações sobre os processos ativos

```
root@DESKTOP-00I5LD3: ~  
root@DESKTOP-00I5LD3:~# ps  
  PID TTY          TIME CMD  
   3 tty1      00:00:00 init  
  81 tty1      00:00:00 su  
  82 tty1      00:00:00 bash  
  92 tty1      00:00:00 vi  
  96 tty1      00:00:00 vi  
 111 tty1      00:00:00 ps  
root@DESKTOP-00I5LD3:~#
```

ps [opções]

- a exibe informações de outros usuários
- u exibe o nome do usuário e a hora de início
- x exibe processos não associados ao terminal
- l exibe linhas detalhadas
- e exibe todos os processos ativos

ps -aux

```
root@DESKTOP-00I5LD3: ~  
127 tty1 00:00:00 ps  
root@DESKTOP-00I5LD3:~# ps -aux  
USER      PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND  
root         1  0.0  0.0   8324    152 ?        Ss   14:38   0:00 /init  
root         3  0.0  0.0   8328    152 tty1     Ss   14:38   0:00 /init  
urubu100     4  0.0  0.0  15220   3740 tty1     S    14:38   0:00 -bash  
urubu100    44  0.0  0.0  15160   3700 tty1     S    14:53   0:00 bash  
urubu100    76  0.0  0.0  24688   6148 tty1     T    15:20   0:00 vi  
urubu100    77  0.0  0.0  24452   5652 tty1     T    15:21   0:00 vi  
root        81  0.0  0.0  14232   1844 tty1     S    15:22   0:00 su root  
root        82  0.0  0.0  13808   2272 tty1     S    15:22   0:00 bash  
root        92  0.0  0.0  24528   5760 tty1     T    15:22   0:00 vi  
root        96  0.0  0.0  24512   5708 tty1     T    15:27   0:00 vi  
root       117  0.0  0.0  24356   5656 tty1     TN   16:29   0:00 vi  
root       119  0.0  0.0  15888   1932 tty1     T    16:30   0:00 top  
root       120  0.0  0.0  24512   5744 tty1     TN   16:36   0:00 vi  
root       121  0.1  0.0  15912   1976 tty1     T    16:36   0:01 top  
root       128  0.0  0.0  15664   1860 tty1     R    17:00   0:00 ps -aux  
root@DESKTOP-00I5LD3:~#
```

REFERÊNCIA BIBLIOGRÁFICA:

TANENBAUM, A. Sistema Operacionais Modernos. Tradução Jorge Ritter. 2ª Edição, São Paulo. Pearson Education do Brasil, 2009.

MACHADO, F. B. Arquitetura de Sistemas Operacionais, 4ª Ed, Rio de Janeiro. LTC, 2007.

SILBERSCHATZ, A. Sistemas Operacionais: Conceitos. 5ª Ed. São Paulo. Prentice Hall, 2000.

- REDHAT. Disponível em www.redhat.com/topics/middleware. Acessado em 19/12/2019.

- FERRARI, F. O Shell. Disponível em <http://www.ferrari.pro.br/home/documents/FFerrari-O-Shell-Unix.pdf>. Acessado em 19/12/2019.

<http://www.agasus.com.br/4-grandes-motivos-para-atualizar-hardware-e-sistemasoperacionais-da-empresa/>

DONDA, D. Windows Power Shell 3.0. Um Guia de Windows PowerShell desenvolvido especificamente para profissionais de infraestrutura. Todo o conteúdo está sob licença da Creative Commons Attribution 3.0 Unported License <http://bit.ly/ZnVDOD>.

Disponível em <http://professorramos.com/Materiais/Documentos/PowerShell%20para%20IT%20Pro%20Book.pdf>. Acessado em 19/12/2019.

LICENÇA MICROSOFT EDUCATION: Instituições de ensino credenciadas, como escolas de ensino fundamental e médio, universidades, faculdades públicas e privadas e faculdades comunitárias estaduais, poderão efetuar o download e reproduzir os Documentos para serem distribuídos em sala de aula. A distribuição fora de sala de aula exigirá permissão por escrito.