


Sistemas Operacionais

Prof. Rodrigo Martins
rodrimartins2005@gmail.com



Cronograma da Aula

- ▶ Gerenciamento de Processos
- ▶ Exercícios

Entendendo Processos no Linux

- Quando executamos algum comando, script ou iniciamos algum programa, o kernel atribui a ele um número de processo (PID) e passa a gerenciar a quantidade de recursos que ele irá disponibilizar para essa atividade. Como haverá sempre diversos processos rodando simultaneamente na máquina o kernel tem uma lista de processos que necessitam de recursos.
- Como não existe atualmente um sistema realmente multitarefa, capaz de realizar diversas atividades realmente ao mesmo tempo, o kernel cria uma fila de processos e a percorre disponibilizando recursos de máquina para cada um deles por um determinado período de tempo. Quanto melhor essa distribuição for efetuada melhor será o desempenho do sistema como um todo e mais próximo de um sistema multitarefas o sistema se parecerá.

Comandos

- Visualizar os processos:
 - `ps -l`

```
rodrigo@rodrigo-VirtualBox:~$ ps -l
```

F	S	UID	PID	PPID	C	PRI	NI	ADDR	SZ	WCHAN	TTY	TIME	CMD
0	S	1000	6945	6936	0	83	3	-	1810	wait	pts/1	00:00:00	bash
0	R	1000	7040	6945	0	83	3	-	1259	-	pts/1	00:00:00	ps

Comandos

- Mostra todos os processos existentes não associados com um terminal.
 - `ps a`

```
root@rodrigo-VirtualBox:/home/rodrigo# ps a
  PID TTY          STAT       TIME COMMAND
  944 tty4      Ss+        0:00   /sbin/getty -8 38400 tty4
  948 tty5      Ss+        0:00   /sbin/getty -8 38400 tty5
  956 tty2      Ss+        0:00   /sbin/getty -8 38400 tty2
  957 tty3      Ss+        0:00   /sbin/getty -8 38400 tty3
  960 tty6      Ss+        0:00   /sbin/getty -8 38400 tty6
 1111 tty1      Ss+        0:00   /sbin/getty -8 38400 tty1
 1243 tty7      Ss+        0:08   /usr/lib/xorg/Xorg -core :0 -seat seat0 -auth /var/ru
 2264 pts/0    Ss         0:00   bash
 2381 pts/0    S          0:00   su
 2382 pts/0    S          0:00   bash
 2393 pts/0    R+         0:00   ps a
root@rodrigo-VirtualBox:/home/rodrigo# exit
exit
rodrigo@rodrigo-VirtualBox:~$ ps a
  PID TTY          STAT       TIME COMMAND
  944 tty4      Ss+        0:00   /sbin/getty -8 38400 tty4
  948 tty5      Ss+        0:00   /sbin/getty -8 38400 tty5
  956 tty2      Ss+        0:00   /sbin/getty -8 38400 tty2
  957 tty3      Ss+        0:00   /sbin/getty -8 38400 tty3
  960 tty6      Ss+        0:00   /sbin/getty -8 38400 tty6
 1111 tty1      Ss+        0:00   /sbin/getty -8 38400 tty1
 1243 tty7      Ss+        0:08   /usr/lib/xorg/Xorg -core :0 -seat seat0 -auth /var/ru
```

Comandos

- Exibe o nome do usuário que iniciou determinado processo e a hora em que isso ocorreu.
 - ps u

```
root@rodrigo-VirtualBox:/home/rodrigo# ps u
USER      PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root      944  0.0  0.1   4656   1856 tty4      Ss+   14:11   0:00 /sbin/getty -8
root      948  0.0  0.1   4656   1948 tty5      Ss+   14:11   0:00 /sbin/getty -8
root      956  0.0  0.1   4656   1868 tty2      Ss+   14:11   0:00 /sbin/getty -8
root      957  0.0  0.1   4656   1968 tty3      Ss+   14:11   0:00 /sbin/getty -8
root      960  0.0  0.1   4656   1876 tty6      Ss+   14:11   0:00 /sbin/getty -8
root     1111  0.0  0.1   4656   2024 tty1      Ss+   14:11   0:00 /sbin/getty -8
root     1243  0.8  5.4 165296 55864 tty7      Ss+   14:11   0:07 /usr/lib/xorg/X
root     2327  0.0  0.3   6404   3552 pts/0    S     14:17   0:00 su
root     2328  0.0  0.3   5684   3536 pts/0    S     14:17   0:00 bash
root     2377  0.0  0.2   5268   2452 pts/0    R+    14:27   0:00 ps u
root@rodrigo-VirtualBox:/home/rodrigo# exit
exit
rodrigo@rodrigo-VirtualBox:~$ ps u
USER      PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
rodrigo   2264  0.0  0.4   7088   4916 pts/0    Ss    14:17   0:00 bash
rodrigo   2379  0.0  0.2   5268   2476 pts/0    R+    14:28   0:00 ps u
rodrigo@rodrigo-VirtualBox:~$
```

Comandos

- Exibe os processos que não estão associados a terminais.
 - `ps x`

```
root@rodrigo-VirtualBox:/home/rodrigo# ps x
  PID TTY          STAT       TIME COMMAND
    1 ?           Ss          0:01 /sbin/init
    2 ?           S            0:00 [kthreadd]
    3 ?           S            0:00 [ksoftirqd/0]
    5 ?           S<           0:00 [kworker/0:0H]
    7 ?           S            0:00 [rcu_sched]
    8 ?           S            0:00 [rcu_bh]
    9 ?           S            0:00 [migration/0]
   10 ?           S            0:00 [watchdog/0]
   11 ?           S            0:00 [kdevtmpfs]
   12 ?           S<           0:00 [netns]
   13 ?           S<           0:00 [perf]
   14 ?           S            0:00 [khungtaskd]
   15 ?           S<           0:00 [writeback]
   16 ?           SN           0:00 [ksmd]
   17 ?           SN           0:00 [khugepaged]
   18 ?           S<           0:00 [crypto]
   19 ?           S<           0:00 [kintegrityd]
   20 ?           S<           0:00 [bioaset]
   21 ?           S<           0:00 [kblockd]
   22 ?           S<           0:00 [ata_sff]
   23 ?           S<           0:00 [md]
```

Comandos

- ps aux

```
root@rodrigo-VirtualBox:/home/rodrigo# ps aux
```

USER	PID	%CPU	%MEM	VSZ	RSS	TTY	STAT	START	TIME	COMMAND
root	1	0.0	0.3	4464	3588	?	Ss	14:11	0:01	/sbin/init
root	2	0.0	0.0	0	0	?	S	14:11	0:00	[kthreadd]
root	3	0.0	0.0	0	0	?	S	14:11	0:00	[ksoftirqd/0]
root	5	0.0	0.0	0	0	?	S<	14:11	0:00	[kworker/0:0H]
root	7	0.0	0.0	0	0	?	S	14:11	0:00	[rcu_sched]
root	8	0.0	0.0	0	0	?	S	14:11	0:00	[rcu_bh]
root	9	0.0	0.0	0	0	?	S	14:11	0:00	[migration/0]
root	10	0.0	0.0	0	0	?	S	14:11	0:00	[watchdog/0]
root	11	0.0	0.0	0	0	?	S	14:11	0:00	[kdevtmpfs]
root	12	0.0	0.0	0	0	?	S<	14:11	0:00	[netns]
root	13	0.0	0.0	0	0	?	S<	14:11	0:00	[perf]
root	14	0.0	0.0	0	0	?	S	14:11	0:00	[khungtaskd]
root	15	0.0	0.0	0	0	?	S<	14:11	0:00	[writeback]
root	16	0.0	0.0	0	0	?	SN	14:11	0:00	[ksmd]
root	17	0.0	0.0	0	0	?	SN	14:11	0:00	[khugepaged]
root	18	0.0	0.0	0	0	?	S<	14:11	0:00	[crypto]
root	19	0.0	0.0	0	0	?	S<	14:11	0:00	[kintegrityd]
root	20	0.0	0.0	0	0	?	S<	14:11	0:00	[bioset]
root	21	0.0	0.0	0	0	?	S<	14:11	0:00	[kblockd]
root	22	0.0	0.0	0	0	?	S<	14:11	0:00	[ata_sff]

Comandos

- Onde os campos são:

USER	- nome do usuário dono do processo;
UID	- número de identificação do usuário dono do processo;
PID	- número de identificação do processo;
PPID	- número de identificação do processo pai;
\%CPU	- porcentagem do processamento usado;
\%MEM	- porcentagem da memória usada;
VSZ	- indica o tamanho virtual do processo;
RSS	- Resident Set Size, indica a quantidade de memória usada (em KB);
TTY	- indica o identificador do terminal do processo;
START	- hora em que o processo foi iniciado;
COMMAND	- nome do comando que executa aquele processo;

Comandos

- Onde os campos são:

PRI - valor da prioridade do processo;

NI - valor preciso da prioridade (geralmente igual aos valores de PRI);

WCHAN - mostra a função do kernel onde o processo se encontra em modo suspenso;

STAT - indica o estado atual do processo, sendo representado por uma letra:

Comandos

- Stat – indica o estado atual do processo, sendo representado por uma letra:
 - **D** Processo morto (usually IO);
 - **R** Running (na fila de processos);
 - **S** Dormindo Interruptamente (aguardando um evento terminar);
 - **T** Parado, por um sinal de controle;
 - **Z** Zombie, terminado mas removido por seu processo pai.

Comandos

- Essas letras podem ser combinadas e ainda acrescidas de:
 - > o processo está rodando com prioridade maior que a padrão, tendo sido definida pelo kernel;
 - < o processo está rodando com prioridade menor que a padrão, tendo sido definida pelo kernel;
 - + o processo é um processo pai, ou seja, possui processos filhos;
 - s o processo é um session leader, ou seja, possui processos que dependem dele;
- **I** o processo possui múltiplas threads;
- **L** o processo possui páginas travadas na memória;
- **N** o processo foi definido com uma prioridade diferente d padrão, tendo sido definida pelo usuário.

Comandos

- Com o top podemos ver o horário atual, quanto tempo a máquina está ligada, quantos usuários estão logados, quantos processos estão em aberto, rodando, em espera e zumbi.

– top

```
top - 14:46:14 up 35 min,  2 users,  load average: 0,10, 0,06, 0,04
Tarefas: 178 total,   1 executando, 177 dormindo,   0 parado,   0 zumbi
%Cpu(s):  7,7 us,  1,7 sy,   0,0 ni, 88,5 id,  2,1 wa,   0,0 hi,   0,0 si,   0,0 st
KiB Mem:  1024412 total,  813868 used,   210544 free,   46272 buffers
KiB Swap:  1046524 total,        0 used,  1046524 free.  331664 cached Mem
```

PID	USUÁRIO	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
1989	rodrigo	20	0	363440	159184	64712	S	6,0	15,5	1:17.32	compiz
1243	root	20	0	165296	55964	27352	S	3,6	5,5	0:19.72	Xorg
2238	rodrigo	20	0	128900	28224	23768	S	1,7	2,8	0:04.21	gnome-termi+
1998	rodrigo	20	0	232752	39528	33848	S	0,7	3,9	0:02.69	nautilus
2451	root	20	0	5504	2808	2412	R	0,7	0,3	0:00.02	top
498	message+	20	0	4924	3192	2224	S	0,3	0,3	0:00.53	dbus-daemon
1781	rodrigo	20	0	126484	28576	20684	S	0,3	2,8	0:00.83	unity-panel+
1894	rodrigo	20	0	38324	9792	7148	S	0,3	1,0	0:00.25	indicator-p+
2070	rodrigo	20	0	83308	22444	16148	S	0,3	2,2	0:00.21	telepathy-i+
2278	rodrigo	20	0	63660	22992	18476	S	0,3	2,2	0:00.13	update-noti+
1	root	20	0	4464	3588	2560	S	0,0	0,4	0:01.27	init
2	root	20	0	0	0	0	S	0,0	0,0	0:00.00	kthreadd
3	root	20	0	0	0	0	S	0,0	0,0	0:00.06	ksoftirqd/0
5	root	0	-20	0	0	0	S	0,0	0,0	0:00.00	kworker/0:0H
7	root	20	0	0	0	0	S	0,0	0,0	0:00.31	rcu_sched
8	root	20	0	0	0	0	S	0,0	0,0	0:00.00	rcu_bh
9	root	rt	0	0	0	0	S	0,0	0,0	0:00.00	migration/0

Comandos

- Onde os campos em CPU(s) são:
 - us = tempo de processamento, executado pelo usuário sy = tempo de processamento, executado pelo sistema ni = tempo de processamento, prioridade alterada pelo usuário id = ociosidade wa = wait - espera de I/O hi = interrupções de hardware si = mapeamento das interrupções pelo Kernel st = steal time

```
top - 14:49:02 up 37 min,  2 users,  load average: 0,02, 0,04, 0,03
Tarefas: 178 total,   1 executando, 177 dormindo,   0 parado,   0 zumbi
%Cpu(s):  4,7 us,  1,7 sy,   0,0 ni, 93,6 id,   0,0 wa,   0,0 hi,   0,0 si,   0,0 st
KiB Mem:  1024412 total,   814144 used,   210268 free,    46308 buffers
KiB Swap: 1046524 total,        0 used,  1046524 free.   331664 cached Mem
```

Comandos

- A primeira linha do comando top também pode ser observada com o comando uptime.
 - uptime

```
root@rodrigo-VirtualBox:/home/rodrigo# uptime
14:49:54 up 38 min,  2 users,  load average: 0,05, 0,05, 0,03
root@rodrigo-VirtualBox:/home/rodrigo#
```

- Onde é apresentado: 16:23:00 - hora atual up 2:14 - tempo que o sistema está ligado 3 users - número de usuários logados
- load average: 0.45, 0.69, 0.74 - média de carga de processamento à 1min atrás, 5 min atrás e à 15min atrás.

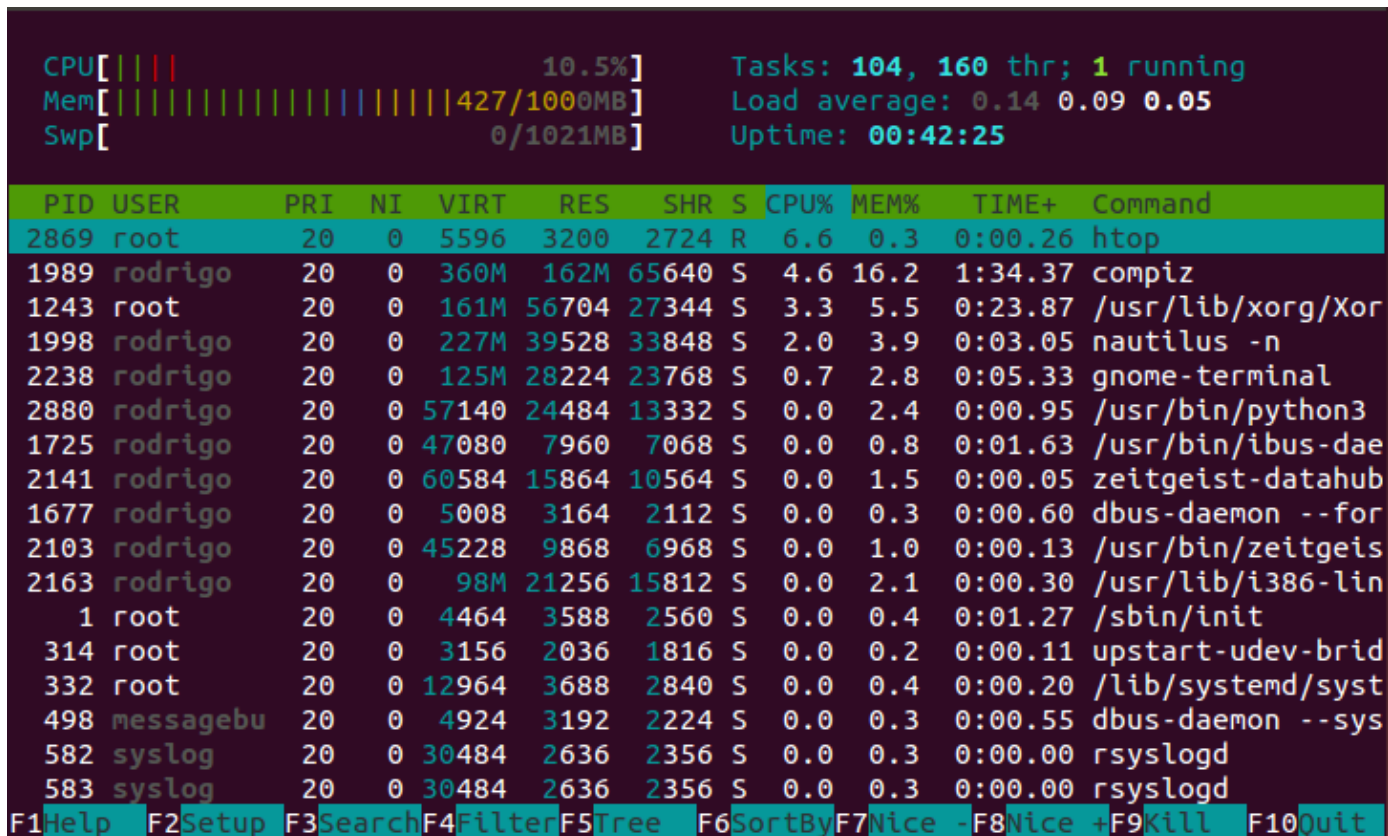
Comandos

- Outro programa que nos ajuda a visualizar os processos é o htop muito mais amigável.
 - htop

```
root@rodrigo-VirtualBox:/home/rodrigo# htop
O programa 'htop' não está instalado no momento. Você pode instalá-lo digitando:
apt-get install htop
root@rodrigo-VirtualBox:/home/rodrigo# apt-get install htop
Lendo listas de pacotes... Pronto
Construindo árvore de dependências
Lendo informação de estado... Pronto
Os NOVOS pacotes a seguir serão instalados:
  htop
0 pacotes atualizados, 1 pacotes novos instalados, 0 a serem removidos e 407 não
atualizados.
```


Comandos

- Outro programa que nos ajuda a visualizar os processos é o htop muito mais amigável.
 - htop



The screenshot shows the htop interface. At the top, system statistics are displayed: CPU usage at 10.5%, memory usage at 427/1000MB, and swap usage at 0/1021MB. System tasks are 104, with 160 threads and 1 running. Load averages are 0.14, 0.09, and 0.05. Uptime is 00:42:25.

PID	USER	PRI	NI	VIRT	RES	SHR	S	CPU%	MEM%	TIME+	Command
2869	root	20	0	5596	3200	2724	R	6.6	0.3	0:00.26	htop
1989	rodrigo	20	0	360M	162M	65640	S	4.6	16.2	1:34.37	compiz
1243	root	20	0	161M	56704	27344	S	3.3	5.5	0:23.87	/usr/lib/xorg/Xor
1998	rodrigo	20	0	227M	39528	33848	S	2.0	3.9	0:03.05	nautilus -n
2238	rodrigo	20	0	125M	28224	23768	S	0.7	2.8	0:05.33	gnome-terminal
2880	rodrigo	20	0	57140	24484	13332	S	0.0	2.4	0:00.95	/usr/bin/python3
1725	rodrigo	20	0	47080	7960	7068	S	0.0	0.8	0:01.63	/usr/bin/ibus-dae
2141	rodrigo	20	0	60584	15864	10564	S	0.0	1.5	0:00.05	zeitgeist-datahub
1677	rodrigo	20	0	5008	3164	2112	S	0.0	0.3	0:00.60	dbus-daemon --for
2103	rodrigo	20	0	45228	9868	6968	S	0.0	1.0	0:00.13	/usr/bin/zeitgeis
2163	rodrigo	20	0	98M	21256	15812	S	0.0	2.1	0:00.30	/usr/lib/i386-lin
1	root	20	0	4464	3588	2560	S	0.0	0.4	0:01.27	/sbin/init
314	root	20	0	3156	2036	1816	S	0.0	0.2	0:00.11	upstart-udev-brid
332	root	20	0	12964	3688	2840	S	0.0	0.4	0:00.20	/lib/systemd/syst
498	messagebu	20	0	4924	3192	2224	S	0.0	0.3	0:00.55	dbus-daemon --sys
582	syslog	20	0	30484	2636	2356	S	0.0	0.3	0:00.00	rsyslogd
583	syslog	20	0	30484	2636	2356	S	0.0	0.3	0:00.00	rsyslogd

At the bottom, function keys are mapped: F1 Help, F2 Setup, F3 Search, F4 Filter, F5 Tree, F6 SortBy, F7 Nice, F8 Nice, F9 Kill, F10 Quit.

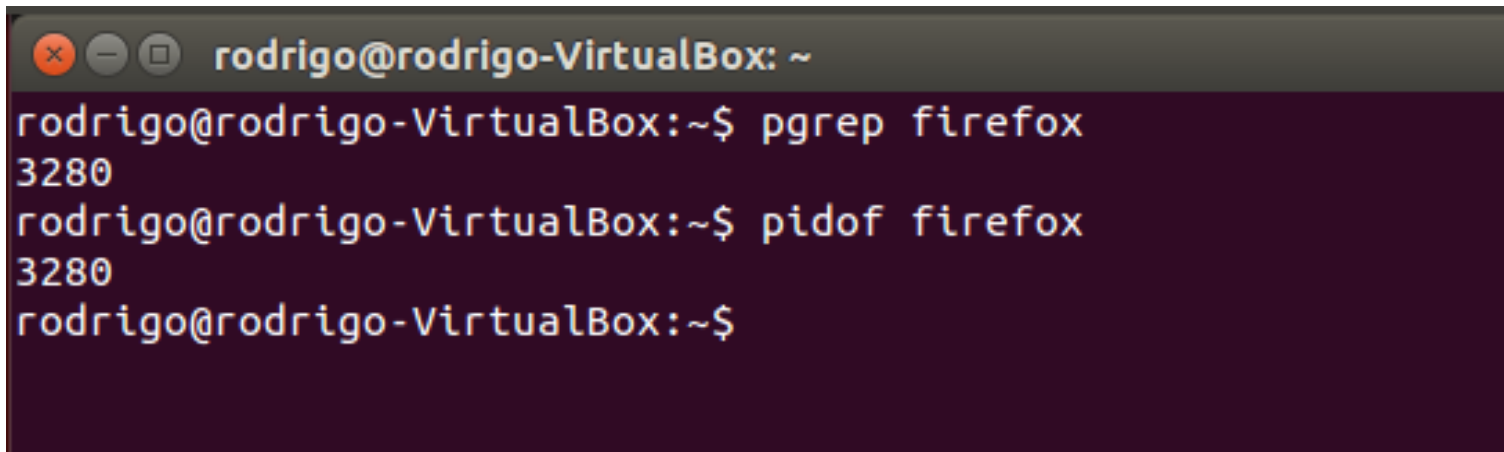
Comandos

- Observe que com o htop você pode navegar na lista de processos. No sistema de processos temos um tipo de processo que se chama threads esses processos são partes de um outro processo. Podemos ver esse cenário em forma de árvore com o comando.
 - pstree

```
root@rodrigo-VirtualBox:/home/rodrigo# pstree
init--ModemManager--2*[{ModemManager}]
    |
    |--NetworkManager--dhclient
    |                   |
    |                   |--dnsmasq
    |                   |
    |                   3*[{NetworkManager}]
    |
    |--accounts-daemon--2*[{accounts-daemon}]
    |
    |--acpid
    |
    |--avahi-daemon--avahi-daemon
    |
    |--bluetoothd
    |
    |--colord--2*[{colord}]
    |
    |--cron
    |
    |--cups-browsed
    |
    |--cupsd
    |
    |--dbus-daemon
    |
    |--6*[getty]
    |
    |--gnome-keyring-d--6*[{gnome-keyring-d}]
    |
    |--kerneloops
    |
    |--lightdm--Xorg
    |           |
    |           |--lightdm--init--at-spi-bus-laun--dbus-daemon
    |           |                   |
    |           |                   |--3*[{at-spi-bus-laun}]
    |           |                   |
    |           |                   |--at-spi2-registr--{at-spi2-registr}
    |           |                   |
    |           |                   |--bamfdaemon--3*[{bamfdaemon}]
```

Comandos

- Um modo fácil de achar o PID de um processo é.
 - psrep firefox
- Ou
 - pidof firefox

A terminal window with a dark purple background and a grey title bar. The title bar contains three window control icons (close, minimize, maximize) and the text 'rodrigo@rodrigo-VirtualBox: ~'. The terminal shows the following commands and output:

```
rodrigo@rodrigo-VirtualBox:~$ pgrep firefox
3280
rodrigo@rodrigo-VirtualBox:~$ pidof firefox
3280
rodrigo@rodrigo-VirtualBox:~$
```

Gerenciando os Processos no Linux

- Apesar do kernel gerenciar os processos, nós podemos enviar sinais a esses processos requisitando que eles alterem seu comportamento. Para isso utilizamos o alguns comandos para enviar um sinal de controle a um determinado processo.
 - **SIGHUP (1)** - Termo Hangup. Desconectar
 - **SIGINT (2)** – Interromper → CTRL + C
 - **SIGKILL (9)** – Terminar imediatamente
 - **SIGTERM (15)** – Terminar de forma “elegante”
 - **SIGCONT (18)** – Continuar a execução
 - **SIGSTOP (19)** – Parar/Pausar a execução → CTRL + Z

Gerenciando os Processos no Linux

- Para ver mais opções.
 - `man signal`

```
SIGNAL(2)                                Linux Programmer's Manual                                SIGNAL(2)

NAME
    signal - ANSI C signal handling

SYNOPSIS
    #include <signal.h>

    typedef void (*sighandler_t)(int);

    sighandler_t signal(int signum, sighandler_t handler);

DESCRIPTION
    The behavior of signal() varies across UNIX versions, and has also varied historically across different versions of Linux. Avoid its use: use sigaction(2) instead. See Portability below.

    signal() sets the disposition of the signal signum to handler, which is either SIG_IGN, SIG_DFL, or the address of a programmer-defined function (a "signal handler").

    If the signal signum is delivered to the process, then one of the following happens:

    Manual page signal(2) line 1 (press h for help or q to quit)
```

- Passando esses sinais aos processos podemos realizar tarefas desde, reiniciar o processo até encerrá-lo de forma forçada.

Gerenciando os Processos no Linux

- O comando **kill**:
 - `kill -sinal PID` → encerra o processo
 - `kill -l número_sinal` → retorna o número do sinal
 - `Killall [opção] nome do programa` → termina processos associados com programas cujos nomes são fornecidos
 - `[-i]` = modo interativo

```
rodrigo@rodrigo-VirtualBox:~$ ps aux
```

USER	PID	%CPU	%MEM	VSZ	RSS	TTY	STAT	START	TIME	COMMAND
root	1	0.0	0.2	4468	2808	?	Ss	09:37	0:01	/sbin/init
root	2	0.0	0.0	0	0	?	S	09:37	0:00	[kthreadd]
root	3	0.0	0.0	0	0	?	S	09:37	0:00	[ksoftirqd/0]
root	5	0.0	0.0	0	0	?	S<	09:37	0:00	[kworker/0:0H]
rodrigo	2132	0.0	0.4	7088	4796	pts/12	Ss	10:29	0:00	bash
rodrigo	2146	14.9	17.3	617484	177788	?	Sl	10:29	0:06	/usr/lib/firefo
rodrigo	2229	0.8	2.1	63728	21536	?	Sl	10:30	0:00	update-notifier

```
rodrigo@rodrigo-VirtualBox:~$ kill -15 2146
rodrigo@rodrigo-VirtualBox:~$ kill -SIGTERM 2146
```

Controle de Tarefas do Shell

- Tarefa é um processo colocado em background;
- As tarefas possuem um número de identificação e começa a contar sempre a partir de 1;
- O PID do processo não tem relação alguma com número da tarefa;
- Normalmente coloca-se um processo em background para liberar o terminal.

```
rodrigo@rodrigo-VirtualBox:~$ firefox
rodrigo@rodrigo-VirtualBox:~$ firefox &          <----- Programa em background
[1] 4686      [número da tarefa] PID
rodrigo@rodrigo-VirtualBox:~$ █
```

Controle de Tarefas

- Exemplo com gedit

```
rodrigo@rodrigo-VirtualBox:~$ pidof gedit
2495
rodrigo@rodrigo-VirtualBox:~$ killall -i gedit
Matar gedit(2495)? (y/N) n
gedit: nenhum processo localizado
rodrigo@rodrigo-VirtualBox:~$ killall gedit
```

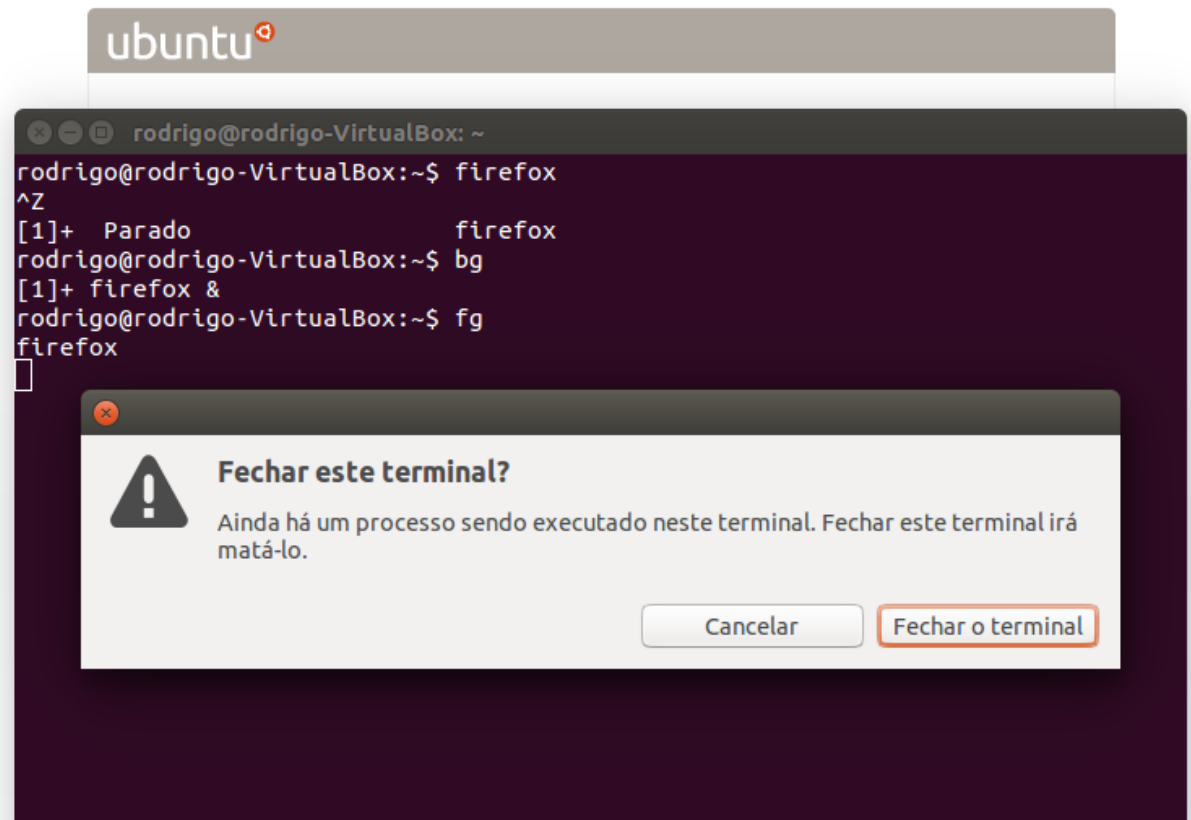
- Exemplo com firefox

```
rodrigo@rodrigo-VirtualBox:~$ firefox
^Z
[1]+  Parado          firefox
rodrigo@rodrigo-VirtualBox:~$ pidof firefox
2726
rodrigo@rodrigo-VirtualBox:~$ kill -18 2726
rodrigo@rodrigo-VirtualBox:~$ kill -9 2726
rodrigo@rodrigo-VirtualBox:~$ firefox
^C
[1]+  Morto           firefox
```

[1]=número da tarefa
+ = tarefa atual

Controle de Tarefas

- Comandos bg e fg
 - bg [número da tarefa]: coloca tarefas em segundo plano (background)
 - fg [número da tarefa]: coloca a tarefa em foreground (tarefa vinculada ao terminal)



Controle de Tarefas

- Comando jobs – Lista as tarefas ativas
 - jobs [opção]
 - -l → Lista também os PIDs

```
rodrigo@rodrigo-VirtualBox:~$ gedit teste.txt
^Z
[1]+  Parado                  gedit teste.txt
rodrigo@rodrigo-VirtualBox:~$ jobs
[1]+  Parado                  gedit teste.txt
rodrigo@rodrigo-VirtualBox:~$ bg
[1]+  gedit teste.txt &
rodrigo@rodrigo-VirtualBox:~$ jobs
[1]+  Executando             gedit teste.txt &
rodrigo@rodrigo-VirtualBox:~$ pidof gedit
5218
rodrigo@rodrigo-VirtualBox:~$ kill -19 5218

[1]+  Parado                  gedit teste.txt
rodrigo@rodrigo-VirtualBox:~$ jobs
[1]+  Parado                  gedit teste.txt
rodrigo@rodrigo-VirtualBox:~$ firefox &
[2] 5228
rodrigo@rodrigo-VirtualBox:~$ kill -19 5228
rodrigo@rodrigo-VirtualBox:~$ jobs
[1]-  Parado                  gedit teste.txt
[2]+  Parado                  firefox
rodrigo@rodrigo-VirtualBox:~$ fg 2
firefox
jobs
rodrigo@rodrigo-VirtualBox:~$ jobs
[1]+  Parado                  gedit teste.txt
rodrigo@rodrigo-VirtualBox:~$ fg 1
gedit teste.txt
```

Prioridades de Processos

- Prioridade do Processo é o tempo relativo de CPU que o Kernel aloca a um processo;
- Coluna PR examinamos a prioridade dos processos;
 - Número PRIORITY = Quanto maior o valor do PR maior é a prioridade do processo.
- Coluna NI ajustamos a prioridade do processo;
 - Número NICE = Quanto menor maior a prioridade do processo. Vai de -20 a +19. Nice padrão é zero.

```
rodrigo@rodrigo-VirtualBox:~$ top
```

```
top - 12:48:11 up 2:20, 2 users, load average: 0,13, 0,10, 0,08
Tarefas: 177 total, 1 executando, 176 dormindo, 0 parado, 0 zumbi
%Cpu(s): 2,0 us, 1,0 sy, 0,0 ni, 97,0 id, 0,0 wa, 0,0 hi, 0,0 si, 0,0 st
KiB Mem: 1024412 total, 796480 used, 227932 free, 20328 buffers
KiB Swap: 1046524 total, 57704 used, 988820 free. 311512 cached Mem
```

PID	USUÁRIO	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
1874	rodrigo	20	0	426888	194892	44464	S	2,3	19,0	10:23.73	compiz
1210	root	20	0	161048	44456	10200	S	1,0	4,3	1:31.54	Xorg
5323	rodrigo	20	0	5504	2856	2460	R	0,7	0,3	0:00.34	top
5175	rodrigo	20	0	129140	32144	23632	S	0,3	3,1	0:02.96	gnome-term+
1	root	20	0	4472	3252	2452	S	0,0	0,3	0:01.80	init
2	root	20	0	0	0	0	S	0,0	0,0	0:00.00	kthreadd

Prioridades de Processos

- Altera-se as prioridades dos processos com os comandos:
 - nice
 - renice
- nice -n ajuste (ajuste é o número nice a ser dado)
 - Ajuste vai de 1 a 19 para usuários normais;
 - E de -20 a 19 para root;
 - Se não for especificado o padrão do ajuste é 10.

```
rodrigo@rodrigo-VirtualBox:~$ firefox
^Z
[1]+  Parado                  firefox
rodrigo@rodrigo-VirtualBox:~$ bg
[1]+  firefox &
rodrigo@rodrigo-VirtualBox:~$ ps -l
F S  UID  PID  PPID  C PRI  NI ADDR SZ WCHAN  TTY          TIME CMD
0 S  1000  5184  5175  0  80   0 -  1810 wait  pts/1      00:00:00 bash
0 S  1000  5342  5184  5  80   0 - 153238 poll s pts/1      00:00:03 firefox
0 R  1000  5399  5184  0  80   0 -  1259 -      pts/1      00:00:00 ps
rodrigo@rodrigo-VirtualBox:~$ ps -l
F S  UID  PID  PPID  C PRI  NI ADDR SZ WCHAN  TTY          TIME CMD
0 S  1000  5184  5175  0  80   0 -  1810 wait  pts/1      00:00:00 bash
0 R  1000  5409  5184  0  80   0 -  1259 -      pts/1      00:00:00 ps
[1]+  Concluído              firefox
rodrigo@rodrigo-VirtualBox:~$
```

Prioridades de Processos

- Diminuindo a prioridade

```
rodrigo@rodrigo-VirtualBox:~$ nice -n 12 firefox &  
[1] 5411  
rodrigo@rodrigo-VirtualBox:~$ ps -l  
F S  UID  PID  PPID  C PRI  NI ADDR SZ WCHAN  TTY          TIME CMD  
0 S  1000  5184  5175  0  80   0  -  1810 wait  pts/1    00:00:00 bash  
0 S  1000  5411  5184  7  92  12  - 155240 poll_s pts/1    00:00:02 firefox  
0 R  1000  5489  5184  0  80   0  -  1259 -      pts/1    00:00:00 ps  
rodrigo@rodrigo-VirtualBox:~$
```

Prioridades de Processos

- nice Negativo

```
rodrigo@rodrigo-VirtualBox:~$ nice -n -16 firefox &
[3] 5732
rodrigo@rodrigo-VirtualBox:~$ nice: não foi possível ajustar o valor de nice: Permissão negada

[3]-  Concluído                nice -n -16 firefox
rodrigo@rodrigo-VirtualBox:~$ sudo nice -n -16 firefox &
[3] 5793
rodrigo@rodrigo-VirtualBox:~$ ps -l
F S  UID  PID  PPID  C PRI  NI ADDR SZ WCHAN  TTY          TIME CMD
0 S  1000  5578  5570  0  80   0 -  1774 wait  pts/1    00:00:00 bash
0 R  1000  5794  5578  0  80   0 -  1259 -      pts/1    00:00:00 ps

[3]+  Parado                  sudo nice -n -16 firefox
rodrigo@rodrigo-VirtualBox:~$ fg
sudo nice -n -16 firefox
[sudo] password for rodrigo:
Sinto muito, tente novamente.
[sudo] password for rodrigo:
rodrigo@rodrigo-VirtualBox:~$ ps -l
F S  UID  PID  PPID  C PRI  NI ADDR SZ WCHAN  TTY          TIME CMD
0 S  1000  5578  5570  0  80   0 -  1774 wait  pts/1    00:00:00 bash
0 R  1000  5853  5578  0  80   0 -  1259 -      pts/1    00:00:00 ps
```

Prioridades de Processos

- Comando renice: muda a prioridade em tempo de execução.
 - Renice +|-novo_numero_nice [opções] alvos
 - -u interpreta alvos como nomes de usuários
 - -p interpreta alvos com PIDs

```
rodrigo@rodrigo-VirtualBox:~$ firefox &
[1] 6201
rodrigo@rodrigo-VirtualBox:~$ ps -l
F S  UID    PID  PPID  C PRI  NI ADDR SZ WCHAN  TTY          TIME CMD
0 S  1000   6189   6180  0  80   0 - 1772 wait  pts/1    00:00:00 bash
0 S  1000   6201   6189 26  80   0 - 155439 poll_s pts/1    00:00:01 firefox
0 R  1000   6256   6189  0  80   0 - 1259 -      pts/1    00:00:00 ps
rodrigo@rodrigo-VirtualBox:~$ renice 15 6201
6201 (ID do processo) old priority 0, new priority 15
rodrigo@rodrigo-VirtualBox:~$ ps -l
F S  UID    PID  PPID  C PRI  NI ADDR SZ WCHAN  TTY          TIME CMD
0 S  1000   6189   6180  0  80   0 - 1772 wait  pts/1    00:00:00 bash
0 S  1000   6201   6189 12  95  15 - 155259 poll_s pts/1    00:00:02 firefox
0 R  1000   6258   6189  0  80   0 - 1259 -      pts/1    00:00:00 ps
rodrigo@rodrigo-VirtualBox:~$ renice 3 -u rodrigo
renice: failed to set priority for 1000 (ID do usuário): Permissão negada
rodrigo@rodrigo-VirtualBox:~$ sudo renice 3 -u rodrigo
[sudo] password for rodrigo:
1000 (ID do usuário) old priority 0, new priority 3
rodrigo@rodrigo-VirtualBox:~$ ps -l
F S  UID    PID  PPID  C PRI  NI ADDR SZ WCHAN  TTY          TIME CMD
0 S  1000   6189   6180  0  83   3 - 1772 wait  pts/1    00:00:00 bash
0 S  1000   6201   6189  2  83   3 - 153810 poll_s pts/1    00:00:04 firefox
0 R  1000   6266   6189  0  83   3 - 1259 -      pts/1    00:00:00 ps
```


Exercícios

- 1- Visualize os processos no linux
- 2 - Mostre os processos que não estão associados a um terminal
- 3 - Mostre o nome do usuário que iniciou um processo
- 4 - Visualize os processos e seus threads
- 5 - Abra e encontre o PID do programa gedit
- 6 - Termine o processo (pelo número do sinal) de forma "elegante"
- 7 - Abra e encontre o PID do firefox
- 8 - Pause a execução do processo (firefox)
- 9 - Continue a execução do processo (firefox)
- 10 - Termine imediatamente o processo firefox (pelo nome do sinal)
- 11 - Abra e pare (usando uma combinação de tecla de atalho) o firefox
- 12 - Termine todos os processos associados do firefox de modo interativo

Exercícios

- 13 - Abra o firefox em background
- 14 - Interrompa o firefox
- 15 - Abra e pare (usando uma combinação de tecla de atalho) o firefox
- 16 - Coloque a tarefa em background
- 17 - Volte a tarefa em foreground
- 18 - Abra pelo terminal o gedit
- 19 - Abra pelo terminal o firefox
- 20 - Liste as tarefas ativas
- 21 - Coloque o gedit em background
- 22 - Liste as tarefas ativas
- 23 - Volte o gedit para foreground
- 24 - Liste as tarefas ativas
- 25 - Abra o firefox e diminua sua prioridade de execução para 95
- 26 - Abra o gedit e aumente sua prioridade de execução para 70

Referências desta Aula

- 4LINUX Essentials

Fim
Obrigado