Encerramento de processos

Controle de processos com sinais

Um *sinal* é uma interrupção de software entregue a um processo. Os sinais relatam eventos a um programa em execução. Os eventos que geram um sinal podem ser um erro, um evento externo (uma solicitação de E/S ou um temporizador expirado) ou um uso explícito de um comando de envio de sinal ou de uma sequência de teclado.

A tabela abaixo lista os principais sinais usados pelos administradores do sistema para o gerenciamento dos processos de rotina. Consulte os sinais por seus nomes abreviados (HUP) ou seus nomes próprios (SIGHUP).

Tabela 8.2. Sinais de gerenciamento de processo fundamentais

Sinal	Name	Definição
1	HUP	`Hangup`: comunica a finalização do processo de controle de um terminal. Também solicita a reinicialização do processo (recarregamento da configuração) sem terminá-lo.
2	INT	`Keyboard interrupt`: faz com que o programa termine. Pode ser bloqueado ou manipulado. Enviado pressionando a combinação de teclas INTR (interrupção) (**Ctrl**+**c**).
3	QUIT	`Keyboard quit`: semelhante a SIGINT, adiciona um despejo do processo ao terminar. Enviado pressionando a combinação de teclas QUIT (**Ctrl**+****).
9	KILL	`Kill, unblockable`: faz com que o programa termine abruptamente. Não pode ser bloqueado, ignorado nem manipulado. É constantemente fatal.
15 *padrão*	TERM	`Terminate`: faz com que o programa termine. Diferente de SIGKILL, pode ser bloqueado, ignorado ou manipulado. A maneira "limpa" de solicitar que um programa seja encerrado. Ele permite que o

		programa conclua as operações essenciais e faça a autolimpeza antes do encerramento.
18	CONT	`Continue`: enviado a um processo para que ele seja retomado se estiver parado. Não pode ser bloqueado. Mesmo se for manipulado, sempre fará o processo continuar.
19	STOP	`Stop, unblockable`: suspende o processo. Não pode ser bloqueado nem manipulado.
20	TSTP	`Keyboard stop`: diferente de SIGSTOP, ele pode ser bloqueado, ignorado ou manipulado. Enviado pressionando a combinação de teclas de suspensão (**Ctrl**+**z**).

Cada sinal tem uma *ação padrão*, que é geralmente uma das seguintes:

- `Term` **: encerrar um programa (sair) imediatamente.**
- `Core` **: salvar uma imagem de memória do programa (despejo de memória) e, em seguida, encerrá-lo.**
- `Stop` **: parar a execução de um programa (suspender) e esperar ele ser continuado (retomado).**

Os programas reagem a sinais de evento esperados pela implementação de rotinas de manipulação para ignorar, substituir ou estender uma ação padrão do sinal.

Enviar sinais por solicitação explícita

Você pode sinalizar o processo em primeiro plano atual digitando uma sequência de controles do teclado para suspender (**Ctrl**+**z**), encerrar (**Ctrl**+**c**) ou despejar a memória (**Ctrl**+****) do processo. No entanto, você pode usar comandos de envio de sinal para enviar sinais a um processo em segundo plano ou a processos em uma sessão diferente.

Você pode especificar sinais pelo nome (por exemplo, com as opções `-HUP` ou `-SIGHUP`) ou pelo número (com a opção `-1` relacionada). Os usuários podem encerrar seus processos, mas privilégios de root são necessários para terminar processos de propriedade do usuário.

O comando `kill` usa um número da PID para enviar um sinal para um processo. Apesar de seu nome, você pode usar o comando `kill` para enviar qualquer sinal, não apenas os sinais para encerrar programas. Você pode usar a opção `-l` do comando `kill` para listar os nomes e números de todos os sinais disponíveis.

```
[user@host ~]$kill -l
 1) SIGHUP       2) SIGINT       3) SIGQUIT      4) SIGILL
5) SIGTRAP
 6) SIGABRT      7) SIGBUS       8) SIGFPE       9) SIGKILL
10) SIGUSR1
11) SIGSEGV     12) SIGUSR2     13) SIGPIPE     14) SIGALRM
15) SIGTERM
16) SIGSTKFLT   17) SIGCHLD     18) SIGCONT     19) SIGSTOP
20) SIGTSTP
...output omitted...
[user@host ~]$ps aux | grep job
5194 0.0 0.1 222448 2980 pts/1 S  16:39 0:00 /bin/bash /hom
e/user/bin/control job1
5199 0.0 0.1 222448 3132 pts/1 S  16:39 0:00 /bin/bash /hom
e/user/bin/control job2
5205 0.0 0.1 222448 3124 pts/1 S  16:39 0:00 /bin/bash /hom
e/user/bin/control job3
5430 0.0 0.0 221860 1096 pts/1 S+ 16:41 0:00 grep --color=a
uto job
[user@host ~]$kill 5194
[user@host ~]$ps aux | grep job
user   5199  0.0  0.1 222448  3132 pts/1    S    16:39   0:
00 /bin/bash /home/user/bin/control job2
user   5205  0.0  0.1 222448  3124 pts/1    S    16:39   0:
00 /bin/bash /home/user/bin/control job3
user   5783  0.0  0.0 221860   964 pts/1    S+   16:43   0:
00 grep --color=auto job
[1]   Terminated              control job1
[user@host ~]$kill -9 5199
[user@host ~]$ps aux | grep job
user   5205  0.0  0.1 222448  3124 pts/1    S    16:39   0:
```

```
00 /bin/bash /home/user/bin/control job3
user   5930  0.0  0.0 221860  1048 pts/1    S+   16:44   0:
00 grep --color=auto job
[2]-  Killed                  control job2
[user@host ~]$kill -SIGTERM 5205
user   5986  0.0  0.0 221860  1048 pts/1  S+  16:45  0:00 g
rep --color=auto job
[3]+  Terminated              control job3
```

Processos específicos de controle

Use o comando `pkill` para sinalizar para um ou mais processos que correspondam aos critérios de seleção. Os critérios de seleção podem ser um nome de comando, um processo pertencente a um usuário específico ou todos os processos do sistema.

Os processos e sessões podem ser sinalizados individualmente ou em conjunto. Para encerrar todos os processos para um usuário, use o comando `pkill`.

Como o processo inicial em uma sessão de login (
líder da sessão) é projetado para lidar com solicitações de término de sessão e para ignorar sinais de teclado não intencionais, encerrar todos os processos e shells de login de usuário requer o sinal SIGKILL.

Primeiro, use o comando
`pgrep` para identificar os números de PID a serem eliminados. Esse comando opera de maneira semelhante ao comando `pkill`, incluindo a maioria das mesmas opções, exceto que o comando `pgrep` lista processos em vez de eliminá-los.

Use o comando
`pgrep` com a opção `-l` para listar os nomes e IDs de processos. Use um dos comandos com a opção `-u` para especificar a ID do usuário proprietário dos processos.

```
[root@host ~]#pgrep -l -u bob
6964 bash
6998 sleep
6999 sleep
7000 sleep
[root@host ~]#pkill -SIGKILL -u bob
[root@host ~]#pgrep -l -u bob
```

Quando os processos que exigem atenção estiverem na mesma sessão de login, ele poderá ser desnecessário para encerrar todos os processos do usuário. Use o comando `w` para determinar o terminal de controle para a sessão. Em seguida, encerre somente os processos que fazem referência à mesma ID de terminal.

A menos que
`SIGKILL` seja especificado, o líder da sessão (aqui, o shell de login `Bash`) lida com a solicitação de término com sucesso e sobrevive a ela, mas todos os outros processos da sessão são terminados.

Use a opção
`-t` para corresponder processos com uma ID de terminal específica.

```
[root@host ~]#pgrep -l -u bob
7391 bash
7426 sleep
7427 sleep
7428 sleep
[root@host ~]#w -u bob
USER     TTY      FROM             LOGIN@   IDLE   JCPU   P
CPU WHAT
bob      tty3                      18:37    5:04   0.03s
0.03s -bash
[root@host ~]#pkill -t tty3
[root@host ~]#pgrep -l -u bob
7391 bash
[root@host ~]#pkill -SIGKILL -t tty3
```

```
[root@host ~]#pgrep -l -u bob
[root@host ~]#
```

O mesmo término seletivo de processos pode ser aplicado usando relações de processos de pai e filho. Use o comando `pstree` para exibir uma árvore de processo ao sistema ou a um único usuário. Use a PID do processo pai para encerrar todos os filhos que foram criados. Desta vez, o shell de login `Bash` pai sobrevive, porque o sinal é orientado apenas a seus processos filhos.

```
[root@host ~]#pstree -p bob
bash(8391)─┬─sleep(8425)
           ├─sleep(8426)
           └─sleep(8427)
[root@host ~]#pkill -P 8391
[root@host ~]#pgrep -l -u bob
bash(8391)
[root@host ~]#pkill -SIGKILL -P 8391
[root@host ~]#pgrep -l -u bob
bash(8391)
```

Sinalização de vários processos

O comando `killall` pode sinalizar vários processos, com base no nome do comando.

```
[user@host ~]$ps aux | grep job
5194  0.0  0.1 222448  2980 pts/1    S    16:39   0:00 /bi
n/bash /home/user/bin/control job1
5199  0.0  0.1 222448  3132 pts/1    S    16:39   0:00 /bi
n/bash /home/user/bin/control job2
5205  0.0  0.1 222448  3124 pts/1    S    16:39   0:00 /bi
n/bash /home/user/bin/control job3
5430  0.0  0.0 221860  1096 pts/1    S+   16:41   0:00 grep
```

```
--color=auto job
[user@host ~]$killall control
[1]   Terminated              control job1
[2]-  Terminated              control job2
[3]+  Terminated              control job3
[user@host ~]$
```

Encerramento de trabalhos em segundo plano

Para encerrar trabalhos em segundo plano, use o comando `kill` e especifique o número do trabalho.

Use o comando `jobs` para localizar o número da tarefa do processo a ser encerrado.

```
[user@host ~]$jobs
[1]-  Running                 sleep 500 &
[2]+  Running                 sleep 1000 &
[user@host ~]$
```

Encerre um trabalho específico usando o comando `kill`. Prefixe o número do trabalho com um sinal de porcentagem (%).

```
[user@host ~]$kill -SIGTERM %1
[user@host ~]$jobs
[2]+  Running                 sleep 1000 &
```

Logout administrativo de usuários

Talvez seja necessário fazer o logout de outros usuários por vários motivos. Alguns possíveis cenários: o usuário cometeu uma violação de segurança; o usuário pode ter usado recursos em excesso; o usuário pode ter um sistema que não responde ou o usuário tem acesso inadequado aos materiais.

Nesses casos, é necessário encerrar a sessão usando sinais administrativamente.

Primeiro, para fazer logoff de um usuário, primeiro identifique a sessão de login a ser finalizada. Use o comando `w` para listar logins de usuários e processos em execução atuais. Observe as colunas `TTY` e `FROM` para determinar as sessões a serem fechadas.

Todas as sessões de login do usuário estão associadas a um dispositivo de terminal (TTY). Se o nome do dispositivo for `pts/N`, ele será um *pseudoterminal* associado a uma janela de terminal gráfica ou uma sessão de login remoto. Se for `tty N`, o usuário estará em um console do sistema, em um console alternativo ou em outro dispositivo do terminal conectado diretamente.

```
[user@host ~]$w
 12:43:06 up 27 min,  5 users,  load average: 0.03, 0.17,
0.66
USER     TTY      FROM             LOGIN@   IDLE   JCPU   P
CPU WHAT
root     tty2                      12:26   14:58   0.04s
0.04s -bash
bob      tty3                      12:28   14:42   0.02s
0.02s -bash
user     pts/1    desktop.example.com 12:41    2.00s  0.03s
0.03s w
[user@host ~]$
```

Descubra há quanto tempo um usuário está no sistema visualizando o tempo de login da sessão. Os recursos da CPU consumidos por tarefas atuais, incluindo tarefas em segundo plano e processos filhos, estão na coluna `JCPU` para cada sessão. O consumo de CPU dos processos em primeiro plano atuais está na coluna `PCPU`.