

Управление процессами

Майоров Дмитрий Андреевич

Содержание

1 Цель работы	5
2 Выполнение лабораторной работы	6
3 Самостоятельная работа	10
3.1 Задание 1	10
3.2 Задание 2	11
4 Выводы	15

Список иллюстраций

2.1	6
2.2	6
2.3	7
2.4	7
2.5	7
2.6	8
2.7	8
2.8	8
2.9	8
2.10	9
3.1	10
3.2	10
3.3	10
3.4	11
3.5	11
3.6	11
3.7	12
3.8	12
3.9	12
3.10	13
3.11	13
3.12	13
3.13	13
3.14	14
3.15	14

Список таблиц

1 Цель работы

Получить навыки управления процессами операционной системы

2 Выполнение лабораторной работы

Получаем полномочия администратора. Вводим нужные команды. Последнюю команду мы запустили без &, поэтому мы пока что не имеем контроль над оболочкой. Нажимаем ctrl+z, чтобы остановить процесс

```
root@mayorovda:~# sleep 3600 &           [1] 3765
root@mayorovda:~# dd if=/dev/zero of=/dev/null &
[2] 3803
root@mayorovda:~# sleep 7200
^Z
[3]+  Stopped                  sleep 7200
```

Рисунок 2.1

Вводим команду jobs. Видим три задания, которые мы запустили

```
root@mayorovda:~# jobs
[1]  Running                  sleep 3600 &
[2]- Running                  dd if=/dev/zero of=/dev/null &
[3]+ Stopped                  sleep 7200
root@mayorovda:~# bg 3
[3]+ sleep 7200 &
```

Рисунок 2.2

Перемещаем первое задание на передний план. Отменяем первое задание с помощью ctrl+c. Проводим аналогичные действия для заданий 2 и 3

```
root@mayorovda:~# fg 1
sleep 3600
^C
root@mayorovda:~# jobs
[2]-  Running                  dd if=/dev/zero of=/dev/null &
[3]+  Running                  sleep 7200 &
root@mayorovda:~# fg 2
dd if=/dev/zero of=/dev/null
^C257344997+0 records in
257344997+0 records out
131760638464 bytes (132 GB, 123 GiB) copied, 79.2569 s, 1.7 GB/s

root@mayorovda:~# fg 3
sleep 7200
^C
```

Рисунок 2.3

В новом терминале делаем новое задание. Закрываем терминал с помощью exit.

```
mayorovda@mayorovda:~$ dd if=/dev/zero of=/dev/null &
[1] 4068
mayorovda@mayorovda:~$ exit
```

Рисунок 2.4

В другом терминале запускаем top. Используем клавишу K, чтобы убить задание dd. Выходим из top

```
root@mayorovda:~# top

top - 14:35:41 up 3 min,  4 users,  load average: 1.52,  0.69,  0.27
Tasks: 257 total,   2 running, 254 sleeping,   0 stopped,   1 zombie
%Cpu(s): 18.1 us, 37.4 sy,  0.3 ni, 43.5 id,  0.0 wa,  0.6 hi,  0.0 si,
MiB Mem : 1961.2 total,    92.0 free, 1436.0 used,   633.7 buff/cac
MiB Swap: 2092.0 total,   1770.0 free,    322.0 used,   525.2 avail Me
```

Рисунок 2.5

Запускаем три новые программы

```
root@mayorovda:~# dd if=/dev/zero of=/dev/null &
[1] 4194
root@mayorovda:~# dd if=/dev/zero of=/dev/null &
[2] 4196
root@mayorovda:~# dd if=/dev/zero of=/dev/null &
[3] 4209
```

Рисунок 2.6

Вводим команду ps aux | grep dd. Она показывает все строки, в которых есть буквы dd

```
/dev/zero of=/dev/null
root      4194 41.5  0.0 226848  2004 pts/0    R    14:36   0:05 dd if=
/dev/zero of=/dev/null
root      4196 35.4  0.0 226848  2008 pts/0    R    14:36   0:04 dd if=
/dev/zero of=/dev/null
root      4209 33.5  0.0 226848  1856 pts/0    R    14:36   0:03 dd if=
/dev/zero of=/dev/null
root      4233  0.0  0.1 227688  2184 pts/0    S+   14:36   0:00 grep -
-color=auto dd
```

Рисунок 2.7

Используем PID первого процесса, чтобы изменить его приоритет

```
root@mayorovda:~# renice -n 5 4194
4194 (process ID) old priority 0, new priority 5
```

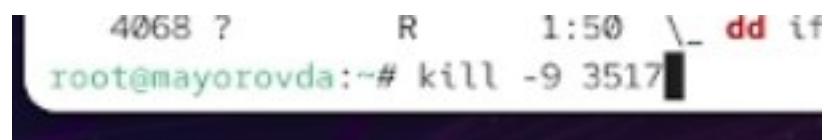
Рисунок 2.8

Вводим ps fax | grep -B5 dd. Параметр -B5 показывает соответствующие запросу строки, включая пять строк до этого

```
3517 ?      Rsl    0:07  \_ /usr/bin/ptyxis --gapplication-service
3524 ?      Ssl    0:00  |  \_ /usr/libexec/ptyxis-agent --socket-f
-3 --rlimit-nofile=1024
3560 pts/0   Ss    0:00  |      \_ /usr/bin/bash
3618 pts/0   S     0:00  |          \_ su -
3659 pts/0   S     0:00  |          \_ -bash
4194 pts/0   RN    0:18  |          \_ dd if=/dev/zero of=/
`v/null
4196 pts/0   R     0:20  |          \_ dd if=/dev/zero of=/
`v/null
4209 pts/0   R     0:19  |          \_ dd if=/dev/zero of=/
`v/null
```

Рисунок 2.9

Находим PID корневой оболочки, из которой были запущены процессы dd, и вводим kill -9 3517. Корневая оболочка закрылась, а вместе с ней и все процессы dd



4068 ? R 1:50 _ dd if
root@mayorovda:~# kill -9 3517

Рисунок 2.10

3 Самостоятельная работа

3.1 Задание 1

Запускаем команду dd if=/dev/zero of=/dev/null трижды как фоновое значение

```
mayorovda@mayorovda:~$ dd if=/dev/zero of=/dev/null &
[1] 4675
mayorovda@mayorovda:~$ dd if=/dev/zero of=/dev/null &
[2] 4677
mayorovda@mayorovda:~$ dd if=/dev/zero of=/dev/null &
[3] 4689
mayorovda@mayorovda:~$
```

Рисунок 3.1

Изменяем приоритет первого процесса на -5, а потом изменяем его же на -15. -15 - это более высокий приоритет

```
mayorovda@mayorovda:~$ sudo renice -5 4675
[sudo] password for mayorovda:
4675 (process ID) old priority 0, new priority -5
```

Рисунок 3.2

```
mayorovda@mayorovda:~$ sudo renice -15 4675
4675 (process ID) old priority -5, new priority -15
```

Рисунок 3.3

Завершаем все процессы

```
mayorovda@mayorovda:~$ pkill -f 'dd if=/dev/zero'
```

Рисунок 3.4

3.2 Задание 2

Запускаем программу yes в фоновом режиме с подавлением потока вывода

```
mayorovda@mayorovda:~$ yes > /dev/null &
[1] 5129
```

Рисунок 3.5

Запускаем программу yes на переднем плане с подавлением потока вывода. Приостановливаем выполнение программы. Заново запускаем программу yes с теми же параметрами, затем завершаем её выполнение

```
mayorovda@mayorovda:~$ yes > /dev/null
^Z
[2]+  Stopped                  yes > /dev/null
mayorovda@mayorovda:~$ fg
yes > /dev/null
^C
```

Рисунок 3.6

Запускаем программу yes на переднем плане без подавления потока вывода. Приостановливаем выполнение программы. Заново запускаем программу yes с теми же параметрами, затем завершаем её выполнение

```
ÿ  
ÿ  
ÿ  
ÿ^Z  
[2]+  Stopped                  yes  
nayorovda@nayorovda:~$
```

Рисунок 3.7

```
ÿ  
ÿ  
ÿ  
ÿ^C  
nayorovda@nayorovda:~$
```

Рисунок 3.8

Проверяем состояния заданий, воспользовавшись командой jobs. Переводим процесс, который у нас выполняется в фоновом режиме, на передний план, затем останавливаем его

```
nayorovda@nayorovda:~$ jobs  
[1]+  Running                  yes > /dev/null &  
nayorovda@nayorovda:~$ fg 1  
yes > /dev/null  
^Z  
[1]+  Stopped                  yes > /dev/null  
nayorovda@nayorovda:~$
```

Рисунок 3.9

Запускаем ещё три программы yes в фоновом режиме с подавлением потока вывода

```
mayorovda@mayorovda:~$ yes > /dev/null &
[2] 5511
mayorovda@mayorovda:~$ yes > /dev/null &
[3] 5526
mayorovda@mayorovda:~$ yes > /dev/null &
```

Рисунок 3.10

Убиваем два процесса: для одного используем его PID, а для другого – его идентификатор конкретного задания

```
mayorovda@mayorovda:~$ kill 5511
[2]  Terminated      yes > /dev/null
mayorovda@mayorovda:~$ kill 3
bash: kill: (3) - Operation not permitted
mayorovda@mayorovda:~$ kill %3
[3]  Terminated      yes > /dev/null
```

Рисунок 3.11

Запускаем ещё несколько программ yes в фоновом режиме с подавлением потока вывода

```
mayorovda@mayorovda:~$ yes > /dev/null &
[51] 5657
```

Рисунок 3.12

Завершаем их работу одновременно, используя команду killall

```
mayorovda@mayorovda:~$ killall yes
```

Рисунок 3.13

Запускаем программу yes в фоновом режиме с подавлением потока вывода. Используя утилиту nice, запускаем программу yes с теми же параметрами и с приоритетом, большим на 5

```
mayorovda@mayorovda:~$ yes > /dev/null &
[2] 5746
mayorovda@mayorovda:~$ nice -n 5 yes > /dev/null &
```

Рисунок 3.14

Используя утилиту renice, изменяем приоритет у одного из потоков yes таким образом, чтобы у обоих потоков приоритеты были равны

```
[2]+ 07:24
mayorovda@mayorovda:~$ sudo renice -5 5746
```

Рисунок 3.15

4 Выводы

Получены навыки управления процессами операционной системы