

Управление процессами

Лабораторная работа №6

Казначеев С.И.

7 октября 2025

Российский университет дружбы народов, Москва, Россия

Информация

:::::::::: {.columns align=center} :: {.column width="70%"}
:::

- Казначеев Сергей Ильич
- Студент
- Российский университет дружбы народов
- [1132240693@pfur.ru] :: {.column width="30%"}
::

Цель работы

Получить навыки управления процессами операционной системы.

Выполнение лабораторной работы

Для начала откроем терминал и перейдем в супер пользователя root после чего введем три команды 1. sleep 3600 & 2. dd if=/dev/zero of=/dev/null & 3. sleep 7200 После остановим процесс sleep 7200

```
[sikaznacheev@localhost ~]$ su -
Пароль:
[root@localhost ~]# sleep 3600 &
[1] 4562
[root@localhost ~]# dd if=/dev/zero of=/dev/null &
[2] 4566
[root@localhost ~]# sleep 7200
^Z
[3]+  Остановлен      sleep 7200
```

Перенос заданий

После чего переводим 1 задание на передний план и отменим ее. Затем проверим статус заданий с помощью команды jobs после проверки проделаем тоже самое с 2 и 3

```
[root@localhost ~]# fg 1
sleep 3600
^C
[root@localhost ~]# jobs
[2]-  Запущен          dd if=/dev/zero of=/dev/null &
[3]+  Запущен          sleep 7200 &
[root@localhost ~]# fg 2
dd if=/dev/zero of=/dev/null
^C329850190+0 записей получено
329850189+0 записей отправлено
168883296768 байт (169 GB, 157 GiB) скопирован, 92,4393 s, 1,8 GB/s

[root@localhost ~]# fg 3
sleep 7200
^C
```

Выход из top

Затем введем команду top чтобы проверить запущена задание dd или нет. После проверки выйдем используя q

top - 12:53:37 up 28 min, 2 users, load average: 2,16, 1,76, 1,07											
Tasks: 304 total, 2 running, 302 sleeping, 0 stopped, 0 zombie											
%Cpu(s): 2,5 us, 3,8 sy, 0,0 ni, 86,8 id, 0,0 wa, 6,9 hi, 0,0 si, 0,0 st											
MiB Mem : 3654,2 total, 696,4 free, 1984,3 used, 1286,8 buff/cache											
MiB Swap: 4040,0 total, 4033,2 free, 6,8 used. 1669,9 avail Mem											
PID to signal/kill [default pid = 4649]											
PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
4649	sikazna+	20	0	220988	1792	1792	R	90,9	0,0	1:10.84	dd
2391	sikazna+	20	10	6467368	443080	146652	S	45,5	11,8	3:21.38	gnome-shell
88	root	39	19	0	0	0	S	4,5	0,0	0:00.61	khugepaged
4234	sikazna+	20	0	3030396	244148	159292	S	4,5	6,5	0:39.15	Isolated Web Co
1	root	20	0	174432	16788	10676	S	0,0	0,4	0:03.12	systemd
2	root	20	0	0	0	0	S	0,0	0,0	0:00.14	kthreadd

Рис. 5: 5

Ввод команд

Теперь перейдем в супер пользователя и введем следующие три команды 1. dd if=/dev/zero of=/dev/null & 2. dd if=/dev/zero of=/dev/null & 3. dd if=/dev/zero of=/dev/null &

```
[root@localhost ~]# su -
[root@localhost ~]# dd if=/dev/zero of=/dev/null &
[1] 4724
[root@localhost ~]# dd if=/dev/zero of=/dev/null &
[2] 4729
[root@localhost ~]#
[root@localhost ~]#
[root@localhost ~]#
[root@localhost ~]# dd if=/dev/zero of=/dev/null &
[3] 4736
```

Просмотр строк dd

Затем введем комаду ps aux | grep dd данная команда нам покажет строки в которых есть буква dd

```
[root@localhost ~]# ps aux | grep dd
root          2  0.0  0.0      0      0 ?          S    12:25   0:00 [kthreadd]
root      1182  0.0  0.0 508432  3584 ?          Sl    12:25   0:00 /usr/sbin/VBoxService --pid
file /var/run/vboxadd-service.sh
sikazna+  2561  0.0  0.8 881864 30328 ?          Ssl   12:39   0:00 /usr/libexec/evolution-addr
essbook-factory
sikazna+  3071  0.0  0.0 232496  1540 ?          S    12:39   0:00 /usr/bin/VBoxClient --dragad
ddrop
sikazna+  3072  0.1  0.0 431300  3076 ?          Sl    12:39   0:01 /usr/bin/VBoxClient --dragad
ddrop
sikazna+  4227  0.0  1.2 269220 46984 ?          Sl    12:45   0:00 /usr/lib64/firefox/firefox
-contentproc -parentBuildID 20250725060637 -sandboxingKind 0 -prefsLen 34161 -prefMapSize 2496
91 -appDir /usr/lib64/firefox/browser {f6fae17a-3016-4822-88c7-fc1ddde5b714} 3961 utility
root      4724 87.6  0.0 220988  1792 pts/0     R    12:54   0:28 dd if=/dev/zero of=/dev/nul
l
root      4729 87.5  0.0 220988  1792 pts/0     R    12:55   0:21 dd if=/dev/zero of=/dev/nul
l
root      4736 88.7  0.0 220988  1792 pts/0     R    12:55   0:12 dd if=/dev/zero of=/dev/nul
l
root      4739  0.0  0.0 221820  2560 pts/0     R+   12:55   0:00 grep --color=auto dd
```

Смена приоритетов

После используем PID ондого из процессов dd чтобы изменить приоритет

```
[root@localhost ~]# renice -n 5 4724
4724 (process ID) old priority 0, new priority 5
```

Рис. 8: 8

Просмотр иерархии

Далее введем `ps fax | grep -B5 dd` и увидим иерархию отношений между процессами и оболочку из которой были запущены процессы `dd` и ее PID

```
2516 ?      Ssl    0:00  \_ /usr/libexec/goa-identity-service
2525 ?      Ssl    0:00  \_ /usr/libexec/evolution-calendar-factory
2528 ?      Ssl    0:00  \_ /usr/libexec/gvfs-udisks2-volume-monitor
2546 ?      Ssl    0:00  \_ /usr/libexec/gvfs-mtp-volume-monitor
2560 ?      Ssl    0:00  \_ /usr/libexec/dconf-service
2561 ?      Ssl    0:00  \_ /usr/libexec/evolution-addressbook-factory

2941 ?      Sl     0:00  \_ /usr/libexec/ibus-x11 --kill-daemon
2948 ?      Ssl    0:00  \_ /usr/libexec/ibus-portal
3041 ?      Ssl    0:00  \_ /usr/libexec/xdg-desktop-portal-gtk
3063 ?      S     0:00  \_ /usr/bin/VBoxClient --seamless
3065 ?      Sl     0:00  |  \_ /usr/bin/VBoxClient --seamless
3071 ?      S     0:00  \_ /usr/bin/VBoxClient --draganddrop
3072 ?      Sl     0:01  |  \_ /usr/bin/VBoxClient --draganddrop

3907 pts/0   Ss    0:00      \_ bash
4507 pts/0   S     0:00      \_ su -
4518 pts/0   S     0:00      \_ -bash
4679 pts/0   S     0:00      \_ su -
4682 pts/0   S     0:00      \_ -bash
4727 pts/0   S     0:00      \_ -bash
```

Убийство процесса

После чего находим PID корневой оболочки из которой были запущены процессы dd и введем
kill -9 4682

```
[root@localhost ~]# kill -9 4682
убито
```

Рис. 10: 10

Запуск команд

Далее запустим команду dd if=/dev/zero of=/dev/null трижды

```
[root@localhost ~]# dd if=/dev/zero of=/dev/null &
[1] 4836
[root@localhost ~]#
[root@localhost ~]#
[root@localhost ~]#
[root@localhost ~]# dd if=/dev/zero of=/dev/null &
[2] 4837
[root@localhost ~]# dd if=/dev/zero of=/dev/null &
[3] 4838
```

Рис. 11: 12

Работа с приоритетами

После чего увеличим приоритет одного из процессов на 5, а потом изменим на 15 и завершим все процессы

```
[root@localhost ~]# renice -n -5 4836
4836 (process ID) old priority 0, new priority -5
[root@localhost ~]# renice -n -15 4836
4836 (process ID) old priority -5, new priority -15
```

Рис. 12: 13

Задание 1-3

После запусти программу yes в фоновом режиме с подавлением потока вывода, далее запустим программу yes на переднем плане с подавлением потока вывода. После чего приостановим процесс и заново запустим с теми же параметрами и завершим ее выполнение

```
[sikaznacheev@localhost ~]$ su -
Пароль:
[root@localhost ~]# yes > /dev/null &
[1] 5056
[root@localhost ~]# yes > /dev/null
^Z
[2]+  Остановлен      yes > /dev/null
[root@localhost ~]# yes > /dev/null
^C
```

Рис. 13: 14

Задание 4

После чего проверим состояния заданий

```
root@localhost ~]# jobs
1]  Запущен      yes > /dev/null &
2]- Остановлен    yes > /dev/null
3]+ Остановлен    yes
```

Рис. 14: 15

Задание 5

Далее переводим процесс который выполняется в фоновом режиме на передний план и остановливем его

```
[root@localhost ~]# fg 1
yes > /dev/null
^Z
[1]+  Остановлен      yes > /dev/null
```

Рис. 15: 16

Задание 6-8

Далее переводим любой процесс с подавлением потока вывода в фоновом режиме, затем проверяем состояние заданий командой jobs и запускаем процесс в фоновом режиме таким образом чтобы он продолжил свою работу даже после отключения

```
[root@localhost ~]# bg 2
[2] yes > /dev/null &
[root@localhost ~]# jobs
[1]+  Остановлен      yes > /dev/null
[2]  Запущен            yes > /dev/null &
[3]-  Остановлен      yes
```

Рис. 16: 17

Задание 9

После закрываем окно и заново запускаем консоль

```
[root@localhost ~]# nohup yes > /dev/null &
[4] 5125
[root@localhost ~]# nohup: ввод игнорируется и поток ошибок перенаправляется на стандартный вывод
[root@localhost ~]#
```

Рис. 17: 18

Задание 10

И проверяем процессы.

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
4724	root	25	5	220988	1792	1792	R	95,7	0,0	22:05.44	dd
4736	root	20	0	220988	1792	1792	R	95,0	0,0	22:12.15	dd
5125	root	20	0	220948	1792	1792	R	95,0	0,0	2:40.73	yes
4729	root	20	0	220988	1792	1792	R	94,7	0,0	22:30.11	dd
5065	root	20	0	220948	1792	1792	R	94,7	0,0	4:56.28	yes
2318	root	20	0	0	0	0	I	0,3	0,0	0:00.45	kworker/u33:0-events_unbound
3072	sikazna+	20	0	431300	3076	2816	S	0,3	0,1	0:01.41	VBoxClient
5064	root	20	0	0	0	0	I	0,3	0,0	0:00.85	kworker/l:2-events
1	root	20	0	174432	18708	10676	S	0,0	0,5	0:03.19	systemd
2	root	20	0	0	0	0	S	0,0	0,0	0:00.15	kthreadd
3	root	20	0	0	0	0	S	0,0	0,0	0:00.00	pool_workqueue_
4	root	0	-20	0	0	0	I	0,0	0,0	0:00.00	kworker/R-rcu_g
5	root	0	-20	0	0	0	I	0,0	0,0	0:00.00	kworker/R-sync_
6	root	0	-20	0	0	0	I	0,0	0,0	0:00.00	kworker/R-slub_
7	root	0	-20	0	0	0	I	0,0	0,0	0:00.00	kworker/R-netns
10	root	20	0	0	0	0	I	0,0	0,0	0:00.00	kworker/u32:0-events_unbound
11	root	0	-20	0	0	0	I	0,0	0,0	0:00.00	kworker/R-mm_pe
12	root	20	0	0	0	0	I	0,0	0,0	0:00.13	kworker/u32:l-netns
13	root	20	0	0	0	0	I	0,0	0,0	0:00.00	rcu_tasks_kthr
14	root	20	0	0	0	0	I	0,0	0,0	0:00.00	rcu_tasks_rude_
15	root	20	0	0	0	0	I	0,0	0,0	0:00.00	rcu_tasks_trace
16	root	20	0	0	0	0	S	0,0	0,0	0:00.14	ksoftirqd/0
17	root	20	0	0	0	0	I	0,0	0,0	0:31.14	rcu_preempt
18	root	20	0	0	0	0	S	0,0	0,0	0:00.00	rcu_exp_par_gp_
19	root	20	0	0	0	0	S	0,0	0,0	0:00.95	rcu_exp_gp_kthr
20	root	rt	0	0	0	0	S	0,0	0,0	0:00.31	migration/0
21	root	-51	0	0	0	0	S	0,0	0,0	0:00.00	idle_inject/0
22	root	20	0	0	0	0	I	0,0	0,0	0:00.43	kworker/0:1-mm_percpu_wq
23	root	20	0	0	0	0	S	0,0	0,0	0:00.00	cpuhp/0
24	root	20	0	0	0	0	S	0,0	0,0	0:00.00	cpuhp/1
25	root	-51	0	0	0	0	S	0,0	0,0	0:00.00	idle_inject/1
26	root	rt	0	0	0	0	S	0,0	0,0	0:00.75	migration/1
27	root	20	0	0	0	0	S	0,0	0,0	0:00.10	ksoftirqd/1
29	root	0	-20	0	0	0	I	0,0	0,0	0:00.00	kworker/l:0H-events_highpri
30	root	20	0	0	0	0	S	0,0	0,0	0:00.00	cpuhp/2
31	root	-51	0	0	0	0	S	0,0	0,0	0:00.00	idle_inject/2

Задание 11

Затем запустим три программы yes в фоновом режиме с подавлением потока вывода

```
[root@localhost ~]# yes > /dev/null &
[5] 5164
[root@localhost ~]# yes > /dev/null &
[6] 5165
[root@localhost ~]#
[root@localhost ~]#
[root@localhost ~]# yes > /dev/null &
[7] 5202
```

Рис. 19: 20

Задание 12

После чего убиваем два процесса для одного используем PID а для другого его индикатор конкретного задания

```
[root@localhost ~]# fg 1
yes > /dev/null
^C
[root@localhost ~]# kill -9 5165
[root@localhost ~]# kill -9 5165
-bash: kill: (5165) - Нет такого процесса
[6]    Убито                  yes > /dev/null
```

Рис. 20: 21

Задание 13

Пробуем послать сигнал 1 процессу запущенному с помощью nohup и обычному процессу

```
[root@localhost ~]# ps aux | grep yes
root      5065 85.5  0.0 220948  1792 pts/1      R     13:09   25:18 yes
root      5077  0.0  0.0 220948  1792 pts/1      T     13:10   0:00 yes
root      5125 96.5  0.0 220948  1792 pts/1      R     13:15   23:07 yes
root      5202 95.3  0.0 220948  1792 pts/1      R     13:22   16:25 yes
sikazna+  5437 93.8  0.0 220948  1792 pts/2      R     13:37   1:36 yes
sikazna+  5450 95.3  0.0 220948  1792 pts/2      R     13:38   1:26 yes
root      5490  0.0  0.0 221688  2432 pts/1      R+    13:39   0:00 grep --color=auto yes
[5]  Обрыв терминальной линии                                         yes > /dev/null
[root@localhost ~]# kill -1 5437
[root@localhost ~]# kill -1 5450
[root@localhost ~]# ps aux | grep yes
root      5065 85.8  0.0 220948  1792 pts/1      R     13:09   26:26 yes
root      5077  0.0  0.0 220948  1792 pts/1      T     13:10   0:00 yes
root      5125 96.5  0.0 220948  1792 pts/1      R     13:15   24:15 yes
root      5202 95.4  0.0 220948  1792 pts/1      R     13:22   17:33 yes
root      5505  0.0  0.0 221688  2432 pts/1      S+    13:40   0:00 grep --color=auto yes
[root@localhost ~]
```

Рис. 21: 22

Задание 14

После чего запускаем еще несколько программ yes в фоновом режиме с подавлением потока вывода

```
[sikaznacheev@localhost ~]$ yes > /dev/null &
[3] 5522
[1]  Обрыв терминальной линии                      yes > /dev/null
[2]  Обрыв терминальной линии                      yes > /dev/null
[sikaznacheev@localhost ~]$ yes > /dev/null &
[4] 5536
```

```
[sikaznacheev@localhost ~]$
```

```
[sikaznacheev@localhost ~]$
```

```
[sikaznacheev@localhost ~]$
```

Задание 15

И убиваем все процессы yes с помощью команды killall yes

```
[s1kaznacheev@localhost ~]$ killall yes
yes(5065): Операция не позволена
yes(5077): Операция не позволена
yes(5125): Операция не позволена
yes(5202): Операция не позволена
[3]  Завершено      yes > /dev/null
[4]- Завершено      yes > /dev/null
[5]+ Завершено      yes > /dev/null
```

Рис. 23: 24

Задание 16-17

Запустим программу yes в фоновом режиме с подавлением потока вывода. После чего используя утилиту nice запустим программу yes с теми же параметрами и с приоритетом большим на 5. И используя утилиту renice изменим приоритет у одного из потоков yes таким образом чтобы у обоих потоков приоритеты были равны

```
[sikaznacheev@localhost ~]$ yes > /dev/null &
[1] 5920
[sikaznacheev@localhost ~]$ nice -n 5 yes > /dev/null &
[2] 5929
[sikaznacheev@localhost ~]$ ps -l | grep yes
0 R 1000 5920 5807 93 80 0 - 55237 - pts/3 00:00:41 yes
0 R 1000 5929 5807 90 85 5 - 55237 - pts/3 00:00:15 yes
[sikaznacheev@localhost ~]$ renice -n 5 5920
5920 (process ID) old priority 0, new priority 5
[sikaznacheev@localhost ~]$ ps -l | grep yes
0 R 1000 5928 5807 94 85 5 - 55237 - pts/3 00:01:43 yes
0 R 1000 5929 5807 94 85 5 - 55237 - pts/3 00:01:16 yes
```

Рис. 24: 25

Контрольный вопрос номер 1

1. Какая команда даёт обзор всех текущих заданий оболочки?

Ответ - команда `jobs` показывает список всех заданий запущенных из текущей оболочки

Контрольный вопрос номер 2

2. Как остановить текущее задание оболочки, чтобы продолжить его выполнение в фоновом режиме?

Ответ - сначала приостанавливаем задание нажав на комбинацию клавиши Ctrl+Z затем отправляем его фон bg

Контрольный вопрос номер 3

3. Какую комбинацию клавиш можно использовать для отмены текущего задания оболочки?

Ответ - комбинацию клавиш Ctrl+C

Контрольный вопрос номер 4

4. Необходимо отменить одно из начатых заданий. Доступ к оболочке, в которой в данный момент работает пользователь, невозможен. Что можно сделать, чтобы отменить задание?

Ответ - используя команду kill указав pid процесса

Контрольный вопрос номер 5

5. Какая команда используется для отображения отношений между родительскими и дочерними процессами?

Ответ - комадна pstree

Контрольный вопрос номер 6

6. Какая команда позволит изменить приоритет процесса с идентификатором 1234 на более высокий?

Ответ - более высокий приоритет = меньшее значение nice sudo. Пример renice -n -5 -p 1234

Контрольный вопрос номер 7

7. В системе в настоящее время запущено 20 процессов dd. Как проще всего остановить их все сразу?

Ответ - команда killall dd

Контрольный вопрос номер 8

8. Какая команда позволяет остановить команду с именем mycommand?

Ответ - команда pkill mycommand

Контрольный вопрос номер 9

9. Какая команда используется в top, чтобы убить процесс?

Ответ - во время работы top нажать на клавишу k

Контрольный вопрос номер 10

10. Как запустить команду с достаточно высоким приоритетом, не рискуя, что не хватит ресурсов для других процессов?

Ответ - запустить команду с пониженным приоритетом. Команда -nice -n 10

Вывод

В результате выполнения лабораторной работы я получил навыки управления процессами операционной системы