

Отчёт по лабораторной работе №6

Управление процессами

Руслан Алиев

Содержание

1	Цель работы	5
2	Выполнение	6
2.1	Управление заданиями	6
2.2	Управление процессами	9
2.3	Задание 1	11
2.4	Задание 2	13
3	Заключение	18
4	Контрольные вопросы	19

Список иллюстраций

2.1	Запуск заданий	7
2.2	Работа команды dd в top	8
2.3	Завершение процесса dd в top	9
2.4	Запуск процессов dd	10
2.5	Просмотр иерархии процессов	11
2.6	Запуск процессов dd	12
2.7	Работа с процессами yes	13
2.8	Проверка статуса заданий	14
2.9	Запуск yes с nohup	15
2.10	Работа yes в top	16
2.11	Сравнение приоритетов процессов yes	17

Список таблиц

1 Цель работы

Получить навыки управления процессами операционной системы.

2 Выполнение

2.1 Управление заданиями

1. Для получения административных прав выполнен переход в режим супер-пользователя с помощью команды

su -

2. Далее были запущены несколько процессов:

sleep 3600 &

dd if=/dev/zero of=/dev/null &

sleep 7200

Первые две команды запустились в фоновом режиме, а третья — в обычном, из-за отсутствия символа **&**.

```

raliev@raliev:~$ su
Password:
root@raliev:/home/raliev# sleep 3600 &
[1] 3447
root@raliev:/home/raliev# dd if=/dev/zero of=/dev/null &
[2] 3476
root@raliev:/home/raliev# sleep 7200
^C
root@raliev:/home/raliev# jobs
[1]-  Running                  sleep 3600 &
[2]+  Running                  dd if=/dev/zero of=/dev/null &
root@raliev:/home/raliev# bg 3
bash: bg: 3: no such job
root@raliev:/home/raliev# sleep 7200
^Z
[3]+  Stopped                  sleep 7200
root@raliev:/home/raliev# jobs
[1]  Running                  sleep 3600 &
[2]-  Running                  dd if=/dev/zero of=/dev/null &
[3]+  Stopped                  sleep 7200
root@raliev:/home/raliev# bg 3
[3]+  sleep 7200 &
root@raliev:/home/raliev# fg 1
sleep 3600
^C
root@raliev:/home/raliev# █

```

Рис. 2.1: Запуск заданий

3. Так как процесс *sleep 7200* занял терминал, он был остановлен комбинацией **Ctrl + Z**.

Проверка активных заданий выполнялась командой **jobs**.

В результате отобразились три задания: два в состоянии *Running* и одно в состоянии *Stopped*.

4. Для возобновления выполнения остановленного задания №3 в фоновом режиме использовалась команда

bg 3

После этого команда **jobs** показала обновлённые статусы процессов.

5. Для возврата задания №1 на передний план была применена команда **fg 1**

Затем процесс был прерван комбинацией **Ctrl + C**.

6. Аналогичным образом были завершены задания №2 и №3.

7. В отдельном терминале, уже под своей учётной записью пользователя, был запущен процесс

dd if=/dev/zero of=/dev/null &

После закрытия терминала командой **exit** процесс продолжал выполняться в системе.

8. Для мониторинга процессов была использована команда **top**.

В списке можно увидеть активное задание **dd**, которое занимает практически 100% процессорного времени.

```
top - 16:44:46 up 9 min, 4 users, load average: 1.00, 0.69, 0.35
Tasks: 260 total, 2 running, 258 sleeping, 0 stopped, 0 zombie
%Cpu(s): 3.4 us, 5.3 sy, 0.1 ni, 91.1 id, 0.0 wa, 0.1 hi, 0.0 si, 0.0 st
MiB Mem : 3909.0 total, 1443.4 free, 1292.8 used, 1409.3 buff/cache
MiB Swap: 4040.0 total, 4040.0 free, 0.0 used, 2616.2 avail Mem
```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
3984	raliev	20	0	226848	1788	1788	R	99.7	0.0	1:52.13	dd
893	root	20	0	18444	9760	7456	S	0.3	0.2	0:00.13	systemd-logind
1187	root	20	0	488244	29164	14316	S	0.3	0.7	0:00.17	tuned
2062	raliev	20	0	4844928	309660	122132	S	0.3	7.7	0:03.77	gnome-shell
3229	raliev	20	0	3020312	348948	99632	S	0.3	8.7	0:02.76	ptxixis
1	root	20	0	49192	41104	10356	S	0.0	1.0	0:01.40	systemd
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	0:00.00	pool_workqueue_release
4	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/R-rcu_gp
5	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/R-sync_wq
6	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/R-slub_flushwq
7	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/R-netns
10	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/0:0H-events_highpri
11	root	20	0	0	0	0	I	0.0	0.0	0:00.00	kworker/u16:0-events_unbound
12	root	20	0	0	0	0	I	0.0	0.0	0:00.04	kworker/u16:1-netns
13	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/R-mm_percpu_wq

Рис. 2.2: Работа команды dd в top

9. Повторный запуск **top** позволил завершить процесс с помощью клавиши **k**, после чего программа была закрыта клавишей **q**.


```
top - 16:44:56 up 9 min, 4 users, load average: 1.00, 0.70, 0.36
Tasks: 259 total, 1 running, 258 sleeping, 0 stopped, 0 zombie
%Cpu(s): 4.0 us, 4.5 sy, 0.0 ni, 91.2 id, 0.0 wa, 0.3 hi, 0.0 st, 0.0 st
MiB Mem : 3909.0 total, 1442.6 free, 1293.5 used, 1409.4 buff/cache
MiB Swap: 4040.0 total, 4040.0 free, 0.0 used, 2615.5 avail Mem
```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
3229	raliev	20	0	3020312	348948	99632	S	4.2	8.7	0:02.99	ptxixis
2062	raliev	20	0	4841856	309660	122132	S	2.5	7.7	0:03.97	gnome-shell
475	root	20	0	0	0	0	I	0.8	0.0	0:00.13	kworker/u17:3-events_unbound
4068	raliev	20	0	231596	5208	3160	R	0.8	0.1	0:00.10	top
1	root	20	0	49192	41104	10356	S	0.0	1.0	0:01.40	systemd
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	0:00.00	pool_workqueue_release
4	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/R-rcu_gp
5	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/R-sync_wq
6	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/R-slub_flushwq
7	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/R-netns
10	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/0:0H-events_highpri
11	root	20	0	0	0	0	I	0.0	0.0	0:00.00	kworker/u16:0-events_unbound
12	root	20	0	0	0	0	I	0.0	0.0	0:00.04	kworker/u16:1-netns
13	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/R-mm_percpu_wq

Рис. 2.3: Завершение процесса dd в top

2.2 Управление процессами

1. Для получения административных прав выполнен переход в режим супер-пользователя с помощью команды

su -

2. Запущено несколько процессов с помощью команды

dd if=/dev/zero of=/dev/null &

Команда была введена трижды, что запустило три отдельных процесса dd в фоновом режиме.

```

root@raliev:/home/raliev#
root@raliev:/home/raliev# dd if=/dev/zero of=/dev/null &
[1] 4581
root@raliev:/home/raliev# dd if=/dev/zero of=/dev/null &
[2] 4583
root@raliev:/home/raliev# dd if=/dev/zero of=/dev/null &
[3] 4585
root@raliev:/home/raliev#
root@raliev:/home/raliev# ps aux | grep dd
root      2  0.0  0.0      0  0 ?        S   16:35   0:00 [kthreadd]
root     93  0.0  0.0      0  0 ?        I<  16:35   0:00 [kworker/R-ipv6_addrconf]
root    1144  0.0  0.0 512956 3008 ?        Sl   16:35   0:00 /usr/sbin/VBoxService --pidfile /var/
run/vboxadd-service.sh
raliev   2572  0.0  0.6 1036404 25380 ?        Ssl  16:36   0:00 /usr/libexec/evolution-addressbook-fa
ctory
root     4581 98.9  0.0 226848 1760 pts/0    R   16:46   0:10 dd if=/dev/zero of=/dev/null
root     4583 98.9  0.0 226848 1808 pts/0    R   16:46   0:09 dd if=/dev/zero of=/dev/null
root     4585 99.4  0.0 226848 1752 pts/0    R   16:46   0:09 dd if=/dev/zero of=/dev/null
root     4612  0.0  0.0 227688 2092 pts/0    S+  16:46   0:00 grep --color=auto dd
root@raliev:/home/raliev# renice -n 5 4581
4581 (process ID) old priority 0, new priority 5
root@raliev:/home/raliev#

```

Рис. 2.4: Запуск процессов dd

3. Для просмотра списка запущенных процессов использовалась команда

ps aux | grep dd

В выводе отобразились все процессы, содержащие в имени *dd*, включая три активных задания.

4. При помощи PID одного из процессов был изменён его приоритет:

renice -n 5

После выполнения команды отображается информация о старом и новом приоритете.

5. Для просмотра иерархии процессов была применена команда

ps fax | grep -B5 dd

В выводе показаны процессы dd вместе с их родительской оболочкой.

```
Process Exited from Signal 9 Restart

--
922 ?      Sns  0:00 /usr/sbin/alsactl -s -n 19 -c -E ALSA_CONFIG_PATH=/etc/alsa/alsactl.conf --
initfile=/lib/alsa/init/00main rdaemon
947 ?      Ssl  0:00 /usr/sbin/ModemManager
949 ?      Ssl  0:00 /usr/bin/python3 -sP /usr/sbin/firewalld --nofork --nopid
951 ?      S   0:00 /usr/sbin/chronyd -F 2
1142 ?     Sl   0:00 /usr/bin/VBoxDRMClient
1144 ?     Sl   0:00 /usr/sbin/VBoxService --pidfile /var/run/vboxadd-service.sh

--
2450 ?     Ssl  0:00 \_ /usr/libexec/evolution-calendar-factory
2459 ?     Ssl  0:00 \_ /usr/libexec/gvfs-goa-volume-monitor
2510 ?     S<sl 0:00 \_ /usr/bin/pipewire
2513 ?     S<sl 0:00 \_ /usr/bin/wireplumber
2514 ?     S<sl 0:00 \_ /usr/bin/pipewire-pulse
2572 ?     Ssl  0:00 \_ /usr/libexec/evolution-addressbook-factory

--
3229 ?     Ssl  0:05 \_ /usr/bin/ptyxis --gaplication-service
3237 ?     Ssl  0:00 | \_ /usr/libexec/ptyxis-agent --socket-fd=3
3307 pts/0  Ss   0:00 | \_ /usr/bin/bash
3351 pts/0  S   0:00 | | \_ su
3394 pts/0  S   0:00 | | \_ bash
4581 pts/0  RN   0:49 | | \_ dd if=/dev/zero of=/dev/null
4583 pts/0  R   0:49 | | \_ dd if=/dev/zero of=/dev/null
4585 pts/0  R   0:48 | | \_ dd if=/dev/zero of=/dev/null
4704 pts/0  R+   0:00 | | \_ ps fax
4705 pts/0  S+   0:00 | | \_ grep --color=auto -B5 dd

root@raliev:/home/raliev# kill -9 3307
Hangup
```

Рис. 2.5: Просмотр иерархии процессов

- Используя PID корневой оболочки, из которой были запущены процессы dd, была выполнена команда

kill -9

В результате оболочка завершилась, а вместе с ней автоматически остановились и все дочерние процессы dd.

2.3 Задание 1

- Сначала был выполнен переход в режим суперпользователя с помощью команды

su -

- Трижды была запущена команда

dd if=/dev/zero of=/dev/null &

В результате созданы три фоновых процесса dd, каждый из которых выпол-

няет бесконечную запись в устройство /dev/null.

```
raliev@raliev:~$
raliev@raliev:~$ su
Password:
root@raliev:/home/raliev# dd if=/dev/zero of=/dev/null &
[1] 4982
root@raliev:/home/raliev# dd if=/dev/zero of=/dev/null &
[2] 4984
root@raliev:/home/raliev# dd if=/dev/zero of=/dev/null &
[3] 4986
root@raliev:/home/raliev# renice -n 5 4982
4982 (process ID) old priority 0, new priority 5
root@raliev:/home/raliev# renice -n 15 4982
4982 (process ID) old priority 5, new priority 15
root@raliev:/home/raliev# killall dd
[1] Terminated dd if=/dev/zero of=/dev/null
[2]- Terminated dd if=/dev/zero of=/dev/null
[3]+ Terminated dd if=/dev/zero of=/dev/null
root@raliev:/home/raliev#
```

Рис. 2.6: Запуск процессов dd

3. Для одного из процессов (PID 4982) приоритет был изменён с 0 на **5** командой

renice -n 5 4982

После этого тот же процесс получил новый приоритет **15** с помощью команды

renice -n 15 4982

Разница заключается в том, что большее положительное значение приоритета означает меньший приоритет выполнения (процесс будет получать меньше процессорного времени по сравнению с другими).

4. Все запущенные процессы dd были завершены одной командой:

killall dd

В результате три фоновых задания перешли в состояние *Terminated*.

2.4 Задание 2

1. Сначала была запущена программа **yes** в фоновом режиме с подавлением потока вывода:

yes > /dev/null &

2. Далее программа **yes** была запущена на переднем плане с теми же параметрами.

Она была приостановлена комбинацией **Ctrl + Z**, а затем возобновлена и завершена.

```
root@raliev:/home/raliev#  
root@raliev:/home/raliev# yes > /dev/null &  
[1] 5280  
root@raliev:/home/raliev# yes > /dev/null  
^Z  
[2]+  Stopped                  yes > /dev/null  
root@raliev:/home/raliev# yes > /dev/null  
^C  
root@raliev:/home/raliev# jobs  
[1]-  Running                  yes > /dev/null &  
[2]+  Stopped                  yes > /dev/null  
root@raliev:/home/raliev#
```

Рис. 2.7: Работа с процессами yes

3. Программа **yes** была также запущена без подавления вывода. После остановки и возобновления выполнение было завершено.
4. Проверка состояний заданий выполнялась командой **jobs**, где видно различие между состояниями *Running* и *Stopped*.

```

root@raliev:/home/raliev#
root@raliev:/home/raliev# jobs
[1]-  Running                  yes > /dev/null &
[2]+  Stopped                  yes > /dev/null
root@raliev:/home/raliev# fg 1
yes > /dev/null
^C
root@raliev:/home/raliev#
root@raliev:/home/raliev# jobs
[2]+  Stopped                  yes > /dev/null
root@raliev:/home/raliev# bg 2
[2]+ yes > /dev/null &
root@raliev:/home/raliev# jobs
[2]+  Running                  yes > /dev/null &
root@raliev:/home/raliev# jobs
[2]+  Running                  yes > /dev/null &
root@raliev:/home/raliev# nohup yes > /dev/null &
[3] 5465
root@raliev:/home/raliev# nohup: ignoring input and redirecting stderr to stdout

root@raliev:/home/raliev# yes > /dev/null &
[4] 5478
root@raliev:/home/raliev# █

```

Рис. 2.8: Проверка статуса заданий

5. Один из процессов был переведён на передний план командой **fg** и остановлен.
6. Другой процесс был возвращён в фоновый режим с помощью **bg**, после чего в списке заданий он отобразился как *Running*.
7. Для запуска процесса, сохраняющего выполнение после закрытия терминала, применена команда:
nohup yes > /dev/null &

```

top - 16:53:10 up 17 min, 5 users, load average: 1.60, 1.42, 0.89
Tasks: 265 total, 4 running, 261 sleeping, 0 stopped, 0 zombie
%Cpu(s): 28.9 us, 51.1 sy, 0.0 ni, 17.8 id, 0.0 wa, 2.2 hi, 0.0 si, 0.0 st
MiB Mem : 3909.0 total, 1401.6 free, 1321.2 used, 1425.9 buff/cache
MiB Swap: 4040.0 total, 4040.0 free, 0.0 used. 2587.8 avail Mem

```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
5293	root	20	0	226820	1764	1764	R	91.7	0.0	0:52.11	yes
5478	root	20	0	226820	1748	1748	R	83.3	0.0	0:20.12	yes
5465	root	20	0	226820	1748	1748	R	75.0	0.0	0:26.05	yes
2062	raliev	20	0	4907280	309928	122144	S	8.3	7.7	0:08.58	gnome-shell
3229	raliev	20	0	3020168	348872	99428	S	8.3	8.7	0:09.54	ptyxis
1	root	20	0	49192	41232	10356	S	0.0	1.0	0:01.82	systemd
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	0:00.00	pool_workqueue_release
4	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/R-rcu_gp
5	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/R-sync_wq
6	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/R-slub_flushwq
7	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/R-netns
10	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/0:0H-events_highpri
11	root	20	0	0	0	0	I	0.0	0.0	0:00.00	kworker/u16:0-events_unbound
12	root	20	0	0	0	0	I	0.0	0.0	0:00.07	kworker/u16:1-netns
13	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/R-mm_percpu_wq
14	root	20	0	0	0	0	I	0.0	0.0	0:00.00	rcu_tasks_kthread
15	root	20	0	0	0	0	I	0.0	0.0	0:00.00	rcu_tasks_rude_kthread

Рис. 2.9: Запуск yes с nohup

- Проверка с помощью **top** подтвердила, что процессы **yes** продолжают выполняться, нагружая процессор.

```

root@raliev:/home/raliev#
root@raliev:/home/raliev# yes > /dev/null &
[1] 5642
root@raliev:/home/raliev# yes > /dev/null &
[2] 5648
root@raliev:/home/raliev# yes > /dev/null &
[3] 5652
root@raliev:/home/raliev# kill 5652
[3]+  Terminated          yes > /dev/null
root@raliev:/home/raliev# fg 2
yes > /dev/null
^C
root@raliev:/home/raliev# kill -1 5642
[1]+  Hangup                yes > /dev/null
root@raliev:/home/raliev# kill -1 5293
root@raliev:/home/raliev# kill -1 5478
root@raliev:/home/raliev# kill -1 5465
root@raliev:/home/raliev# yes > /dev/null &
[1] 5844
root@raliev:/home/raliev# yes > /dev/null &
[2] 5846
root@raliev:/home/raliev# yes > /dev/null &
[3] 5848
root@raliev:/home/raliev# killall yes
[1]  Terminated          yes > /dev/null
[2]-  Terminated          yes > /dev/null
[3]+  Terminated          yes > /dev/null
root@raliev:/home/raliev#

```

Рис. 2.10: Работа yes в top

9. В фоновом режиме были запущены ещё три процесса **yes**. Два из них были завершены разными способами: один по PID, другой по номеру задания.
10. Для проверки работы сигналов был отправлен сигнал **SIGHUP (1)** обычному процессу и процессу, запущенному с **nohup**. Первый процесс завершился, второй продолжил выполнение.
11. Вновь было запущено несколько процессов **yes**. Все они были завершены одновременно командой **killall yes**.
12. Для запуска процесса с пониженным приоритетом была использована команда:
nice -n 5 yes > /dev/null &

Второй процесс был запущен с обычным приоритетом для сравнения. Проверка с помощью **ps -l** показала разницу в значениях поля NI (nice).

```

root@raliev:/home/raliev#
root@raliev:/home/raliev# yes > /dev/null &
[1] 5928
root@raliev:/home/raliev# nice -n 5 yes > /dev/null &
[2] 5951
root@raliev:/home/raliev# ps -l
F S  UID      PID     PPID  C PRI  NI ADDR SZ WCHAN  TTY          TIME CMD
4 S   0       5521    4014   0  80   0 - 58153 do_wai pts/2      00:00:00 su
4 S   0       5543    5521   0  80   0 - 57575 do_wai pts/2      00:00:00 bash
4 R   0       5928    5543  99  80   0 - 56705 -      pts/2      00:00:15 yes
4 R   0       5951    5543  99  85   5 - 56705 -      pts/2      00:00:03 yes
4 R   0       5963    5543   0  80   0 - 57682 -      pts/2      00:00:00 ps
root@raliev:/home/raliev# renice -n 5 5928
5928 (process ID) old priority 0, new priority 5
root@raliev:/home/raliev# ps -l
F S  UID      PID     PPID  C PRI  NI ADDR SZ WCHAN  TTY          TIME CMD
4 S   0       5521    4014   0  80   0 - 58153 do_wai pts/2      00:00:00 su
4 S   0       5543    5521   0  80   0 - 57575 do_wai pts/2      00:00:00 bash
4 R   0       5928    5543  99  85   5 - 56705 -      pts/2      00:00:34 yes
4 R   0       5951    5543  99  85   5 - 56705 -      pts/2      00:00:22 yes
4 R   0       6004    5543   0  80   0 - 57682 -      pts/2      00:00:00 ps
root@raliev:/home/raliev# killall yes
[1]-  Terminated          yes > /dev/null
[2]+  Terminated          nice -n 5 yes > /dev/null
root@raliev:/home/raliev#

```

Рис. 2.11: Сравнение приоритетов процессов yes

13. С помощью утилиты **renice** приоритет одного из процессов был изменён до **5**, что подтвердилось выводом команды **ps -l**.

3 Заключение

В ходе работы были изучены основные способы управления заданиями и процессами в Linux.

Были рассмотрены приёмы запуска процессов на переднем и фоновом режиме, их приостановка, возобновление и завершение.

Особое внимание уделялось использованию команд **jobs**, **fg**, **bg**, **kill**, а также инструментов **nice** и **renice** для управления приоритетами.

Дополнительно была изучена работа с утилитой **top** для мониторинга процессов и их завершения.

4 Контрольные вопросы

1. Какая команда даёт обзор всех текущих заданий оболочки?

`jobs`

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

Нажать **Ctrl + Z**, затем выполнить команду **bg %номер_задания**

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

Ctrl + C

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

Использовать команду **kill** или **killall** в другой оболочке

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

`ps fax`

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

`renice -n -5 -p 1234`

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

killall dd

8. **Какая команда позволяет остановить команду с именем mycommand?**

killall mycommand

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

Внутри top нажать клавишу **k**, ввести PID процесса и подтвердить

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

Использовать **nice** с положительным значением (например, nice -n 10 mycommand)