

Отчёт по лабораторной работе №6

Управление процессами

Шаханеоядж Хаоладар

Содержание

1	Цель работы	5
2	Выполнение	6
3	Выполнение	7
3.1	Управление заданиями	7
3.2	Управление процессами	10
3.3	Задание 1	12
3.4	Задание 2	13
4	Контрольные вопросы	18
5	Заключение	20

Список иллюстраций

3.1	Продолжение выполнения job в фоне	8
3.2	Перевод процессов на передний план и завершение	8
3.3	Фоновый процесс в другом терминале	9
3.4	Просмотр процессов через top	9
3.5	Завершение процесса dd через top	10
3.6	Запуск фоновых процессов dd	11
3.7	Завершение процессов через kill	12
3.8	Изменение приоритета процесса	13
3.9	Работа yes на переднем плане и завершение	14
3.10	Перевод процесса на передний план	14
3.11	Вывод top с процессами yes	15
3.12	Завершение процессов по PID и job ID	16
3.13	Сравнение приоритетов процессов	17
3.14	Изменение приоритетов с помощью renice	17

Список таблиц

1 Цель работы

Получить навыки управления процессами операционной системы.

2 Выполнение

3 Выполнение

3.1 Управление заданиями

1. Для начала работы были получены полномочия администратора с помощью команды **su -**.
2. Далее запущены процессы:
 - **sleep 3600 &** — задание 1 (ожидание в течение часа в фоновом режиме);
 - **dd if=/dev/zero of=/dev/null &** — задание 2 (копирование потока нулей в /dev/null в фоне);
 - **sleep 7200** — задание 3 (ожидание 2 часа в режиме переднего плана).
3. Поскольку третья команда была запущена без **&**, оболочка была заблокирована. С помощью комбинации **Ctrl+Z** процесс был остановлен (статус *Stopped*).
4. Для просмотра списка заданий выполнена команда **jobs**, отобразившая три задания: два выполнялись в фоновом режиме, одно находилось в состоянии *Stopped*.
5. Задание 3 было переведено в фоновый режим командой **bg 3**. Повторная проверка через **jobs** показала, что все процессы находятся в состоянии *Running*.

```

haoladar@haoladar:~$ su
Password:
root@haoladar:/home/haoladar# sleeo 3600 &
[1] 3401
root@haoladar:/home/haoladar# bash: sleeo: command not found...

[1]+  Exit 127                  sleeo 3600
root@haoladar:/home/haoladar# sleep 3600 &
[1] 3429
root@haoladar:/home/haoladar# dd if=/dev/zero of=/dev/null &
[2] 3470
root@haoladar:/home/haoladar# sleep 7200
^Z
[3]+  Stopped                  sleep 7200
root@haoladar:/home/haoladar# jobs
[1]  Running                  sleep 3600 &
[2]-  Running                  dd if=/dev/zero of=/dev/null &
[3]+  Stopped                  sleep 7200
root@haoladar:/home/haoladar# bg 3
[3]+  sleep 7200 &
root@haoladar:/home/haoladar# jobs
[1]  Running                  sleep 3600 &
[2]-  Running                  dd if=/dev/zero of=/dev/null &
[3]+  Running                  sleep 7200 &
root@haoladar:/home/haoladar# █

```

Рис. 3.1: Продолжение выполнения job в фоне

6. Для возврата задания 1 на передний план применена команда **fg 1**. Процесс был прерван комбинацией **Ctrl+C**. Аналогичным образом завершены задания 2 и 3.

```

root@haoladar:/home/haoladar#
root@haoladar:/home/haoladar# fg 1
sleep 3600
^C
root@haoladar:/home/haoladar# fg 2
dd if=/dev/zero of=/dev/null
^C108624478+0 records in
108624478+0 records out
55615732736 bytes (56 GB, 52 GiB) copied, 73.6234 s, 755 MB/s
root@haoladar:/home/haoladar# fg 3
sleep 7200
^C
root@haoladar:/home/haoladar# jobs
root@haoladar:/home/haoladar# █

```

Рис. 3.2: Перевод процессов на передний план и завершение

7. В отдельном терминале под пользователем была запущена команда:
dd if=/dev/zero of=/dev/null &

Процесс успешно стартовал и продолжил выполнение в фоне.

```
haoladar@haoladar:~$ dd if=/dev/zero of=/dev/null &
[1] 3778
haoladar@haoladar:~$ █
```

Рис. 3.3: Фоновый процесс в другом терминале

8. После закрытия терминала команда **top** показала, что процесс **dd** всё ещё работает от имени пользователя.

```
top - 14:23:41 up 5 min, 4 users, load average: 0.72, 0.39, 0.17
Tasks: 263 total, 2 running, 261 sleeping, 0 stopped, 0 zombie
%Cpu(s): 5.7 us, 5.7 sy, 0.0 ni, 85.7 id, 0.0 wa, 2.9 hi, 0.0 si, 0.0 st
MiB Mem : 3909.0 total, 1368.5 free, 1365.0 used, 1412.2 buff/cache
MiB Swap: 4040.0 total, 4040.0 free, 0.0 used, 2543.9 avail Mem
```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
3778	haoladar	20	0	226848	1896	1896	R	100.0	0.0	0:23.42	dd
1	root	20	0	49192	41272	10256	S	0.0	1.0	0:01.46	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-slab_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	0:00.02	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
16	root	20	0	0	0	0	I	0.0	0.0	0:00.00	rcu_tasks_trace_kthread
17	root	20	0	0	0	0	S	0.0	0.0	0:00.01	ksoftirqd/0
18	root	20	0	0	0	0	I	0.0	0.0	0:00.09	rcu_preempt
19	root	20	0	0	0	0	S	0.0	0.0	0:00.00	rcu_exp_par_gp_kthread_worker/0
20	root	20	0	0	0	0	S	0.0	0.0	0:00.08	rcu_exp_gp_kthread_worker
21	root	rt	0	0	0	0	S	0.0	0.0	0:00.11	migration/0
22	root	-51	0	0	0	0	S	0.0	0.0	0:00.00	idle_inject/0
23	root	20	0	0	0	0	S	0.0	0.0	0:00.00	cpuhp/0

Рис. 3.4: Просмотр процессов через top

9. Для завершения работы процесса в **top** была использована команда **k**. После подтверждения PID процесс был завершён.

```
top - 14:25:09 up 7 min, 4 users, load average: 0.94, 0.55, 0.25
Tasks: 260 total, 1 running, 259 sleeping, 0 stopped, 0 zombie
%Cpu(s): 4.7 us, 6.3 sy, 0.1 ni, 88.7 id, 0.0 wa, 0.2 hi, 0.0 si, 0.0 st
MiB Mem : 3909.0 total, 1321.8 free, 1410.6 used, 1413.4 buff/cache
MiB Swap: 4040.0 total, 4040.0 free, 0.0 used, 2498.4 avail Mem
```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
3199	haoladar	20	0	3025836	264508	99556	S	2.0	6.6	0:03.22	ptyxis
2134	haoladar	20	0	5051512	323472	122096	S	1.0	8.1	0:04.13	gnome-shell
597	root	20	0	0	0	0	S	0.2	0.0	0:00.21	xfsaild/dm-0
1989	root	20	0	0	0	0	I	0.2	0.0	0:00.06	kworker/u19:3-events_unbound
1152	root	20	0	574184	2520	2392	S	0.1	0.1	0:00.30	VBoxDRMClient
2003	haoladar	20	0	23160	14256	10160	S	0.1	0.4	0:00.18	systemd
2500	haoladar	20	0	614536	10176	8768	S	0.1	0.3	0:00.04	goa-identity-se
2917	root	20	0	717132	22296	17688	S	0.1	0.6	0:00.06	fwupd
3207	haoladar	20	0	377744	6424	5912	S	0.1	0.2	0:00.09	ptyxis-agent
1	root	20	0	49192	41272	10256	S	0.0	1.0	0:01.52	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	0:00.03	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. 3.5: Завершение процесса dd через top

3.2 Управление процессами

1. Для начала были получены полномочия администратора с помощью команды **su -**.
2. Запущены три фоновых процесса:
 - **dd if=/dev/zero of=/dev/null &** — процесс 1
 - **dd if=/dev/zero of=/dev/null &** — процесс 2
 - **dd if=/dev/zero of=/dev/null &** — процесс 3

```
haoladar@haoladar:~$ su
Password:
root@haoladar:/home/haoladar# dd if=/dev/zero of=/dev/null &
[1] 4384
root@haoladar:/home/haoladar# dd if=/dev/zero of=/dev/null &
[2] 4386
root@haoladar:/home/haoladar# dd if=/dev/zero of=/dev/null &
[3] 4388
root@haoladar:/home/haoladar# ps aux | grep dd
root      2  0.0  0.0      0  0 ?        S   14:17   0:00 [kthreadd]
root     91  0.0  0.0      0  0 ?        I<  14:17   0:00 [kworker/R-ipv6_addrconf]
root    1154  0.0  0.0 578492 3024 ?        Sl  14:17   0:00 /usr/sbin/VBoxService --pidfile /var/run/
vboxadd-service.sh
haoladar 2533  0.0  0.6 962676 25360 ?        Ssl 14:19   0:00 /usr/libexec/evolution-addressbook-factor
y
root    4384 99.2  0.0 226848 1792 pts/2    R   14:26   0:12 dd if=/dev/zero of=/dev/null
root    4386 99.0  0.0 226848 1900 pts/2    R   14:26   0:11 dd if=/dev/zero of=/dev/null
root    4388 99.2  0.0 226848 1728 pts/2    R   14:26   0:10 dd if=/dev/zero of=/dev/null
root    4424  0.0  0.0 227688 2140 pts/2    S+  14:26   0:00 grep --color=auto dd
root@haoladar:/home/haoladar#
```

Рис. 3.6: Запуск фоновых процессов dd

3. Для проверки списка запущенных процессов использована команда **ps aux | grep dd**.

В выводе показаны все строки, содержащие dd. Три последних строки соответствуют активным процессам dd, запущенным ранее.

4. С помощью команды **renice -n 5 <PID>** можно изменить приоритет выполнения одного из процессов dd, задав новое значение.

5. Для анализа иерархии процессов выполнена команда **ps fax | grep -B5 dd**. Параметр -B5 добавил пять строк выше совпадения, что позволило отобразить дерево процессов и определить родительскую оболочку, из которой были запущены процессы dd.

6. Был определён PID оболочки, породившей процессы dd.

Для завершения всех связанных процессов выполнена команда **kill -9 <PID>**.

В результате оболочка завершила работу, а вместе с ней были остановлены и все дочерние процессы dd.

```
Process Exited from Signal 9 Restart

959 ?      Ssl  0:00 /usr/sbin/ModemManager
960 ?      Ssl  0:00 /usr/bin/python3 -sP /usr/sbin/firewalld --nofork --nopic
1152 ?     Sl   0:00 /usr/bin/VBoxDRMClient
1154 ?     Sl   0:00 /usr/sbin/VBoxService --pidfile /var/run/vboxadd-service.sh

--
2462 ?     Ssl  0:00 \_ /usr/libexec/gvfs-mtp-volume-monitor
2483 ?     Ssl  0:00 \_ /usr/libexec/evolution-calendar-factory
2484 ?     Ssl  0:00 \_ /usr/libexec/gvfs-gphoto2-volume-monitor
2500 ?     Ssl  0:00 \_ /usr/libexec/goa-identity-service
2510 ?     Ssl  0:00 \_ /usr/libexec/gvfs-goa-volume-monitor
2533 ?     Ssl  0:00 \_ /usr/libexec/evolution-addressbook-factory

--
3319 pts/0  S    0:00 |      | \_ su
3351 pts/0  S+   0:00 |      | \_ bash
3816 pts/2  Ss   0:00 |      \_ /usr/bin/bash
4295 pts/2  S    0:00 |      \_ su
4320 pts/2  S    0:00 |      \_ bash
4384 pts/2  RN   0:58 |      \_ dd if=/dev/zero of=/dev/null
4386 pts/2  R    0:57 |      \_ dd if=/dev/zero of=/dev/null
4388 pts/2  R    0:57 |      \_ dd if=/dev/zero of=/dev/null
4527 pts/2  R+   0:00 |      \_ ps fax
4528 pts/2  S+   0:00 |      \_ grep --color=auto -B5 dd

root@haoladar:/home/haoladar# kill -9 3816
root@haoladar:/home/haoladar#
Hangup
█
```

Рис. 3.7: Завершение процессов через kill

3.3 Задание 1

1. Были запущены три фоновых процесса с помощью команды:

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

В результате для каждого процесса был назначен собственный PID.

2. Приоритет одного из процессов был изменён с помощью команды **renice -n -5 <PID>**.

Это позволило увеличить его приоритет (чем меньше значение, тем выше приоритет).

```

haoladar@haoladar:~$
haoladar@haoladar:~$ dd if=/dev/zero of=/dev/null &
[1] 5006
haoladar@haoladar:~$ dd if=/dev/zero of=/dev/null &
[2] 5008
haoladar@haoladar:~$ dd if=/dev/zero of=/dev/null &
[3] 5010
haoladar@haoladar:~$ renice -n 5 5006
5006 (process ID) old priority 0, new priority 5
haoladar@haoladar:~$ renice -n 15 5006
5006 (process ID) old priority 5, new priority 15
haoladar@haoladar:~$ 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
haoladar@haoladar:~$

```

Рис. 3.8: Изменение приоритета процесса

3. Приоритет того же процесса был изменён повторно с помощью команды **renice -n 15 <PID>**, что понизило его приоритет по сравнению с другими.
4. Для завершения всех процессов dd использована команда **killall dd**.
Все три фоновых процесса были корректно завершены.

3.4 Задание 2

1. Запущена программа **yes** в фоновом режиме с подавлением потока вывода:
yes > /dev/null &
2. Программа **yes** была запущена на переднем плане с перенаправлением вывода.
Сначала её выполнение приостановлено комбинацией **Ctrl+Z**, затем возобновлено, и процесс был завершён с помощью **Ctrl+C**.

```

haoladar@haoladar:~$ su
Password:
root@haoladar:/home/haoladar# yes > /dev/null &
[1] 5312
root@haoladar:/home/haoladar# yes > /dev/null
^Z
[2]+  Stopped                  yes > /dev/null
root@haoladar:/home/haoladar# yes > /dev/null
^C
root@haoladar:/home/haoladar# jobs
[1]-  Running                  yes > /dev/null &
[2]+  Stopped                  yes > /dev/null
root@haoladar:/home/haoladar#

```

Рис. 3.9: Работа yes на переднем плане и завершение

3. Аналогичные действия были выполнены с программой **yes** без перенаправления вывода: приостановка (Ctrl+Z), возобновление и завершение (Ctrl+C).
4. Для проверки состояний процессов использована команда **jobs**.
Отображены как выполняющиеся (*Running*), так и остановленные (*Stopped*) задания.
5. Один из фоновых процессов был выведен на передний план с помощью **fg** **<номер>** и завершён.

```

root@haoladar:/home/haoladar# jobs
root@haoladar:/home/haoladar# jobs
[1]-  Running                  yes > /dev/null &
[2]+  Stopped                  yes > /dev/null
root@haoladar:/home/haoladar# fg 1
yes > /dev/null
^C
root@haoladar:/home/haoladar# jobs
[2]+  Stopped                  yes > /dev/null
root@haoladar:/home/haoladar# bg 2
[2]+  yes > /dev/null &
root@haoladar:/home/haoladar# jobs
[2]+  Running                  yes > /dev/null &
root@haoladar:/home/haoladar# nohup yes > /dev/null &
[3] 5508
root@haoladar:/home/haoladar# nohup: ignoring input and redirecting stderr to stdout

root@haoladar:/home/haoladar# jobs
[2]-  Running                  yes > /dev/null &
[3]+  Running                  nohup yes > /dev/null &
root@haoladar:/home/haoladar#

```

Рис. 3.10: Перевод процесса на передний план

6. Остановленный процесс был возвращён в фоновый режим командой **bg** **<номер>**. Его статус изменился на *Running*.

7. Для запуска процесса, который продолжит работу после выхода из терминала, использована команда:

```
nohup yes > /dev/null &
```

Это позволило игнорировать сигнал SIGHUP.

8. С помощью команды **top** просмотрен список процессов. В нём видно, что **yes** активно загружает CPU.

```
top - 14:35:08 up 17 min, 5 users, load average: 1.14, 0.85, 0.59
Tasks: 260 total, 3 running, 257 sleeping, 0 stopped, 0 zombie
%Cpu(s): 18.9 us, 29.7 sy, 0.0 ni, 51.4 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st
MiB Mem : 3909.0 total, 1356.6 free, 1364.2 used, 1428.4 buff/cache
MiB Swap: 4040.0 total, 4040.0 free, 0.0 used, 2544.8 avail Mem
```

	PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
	5324	root	20	0	226820	1676	1676	R	100.0	0.0	0:49.73	yes
	5508	root	20	0	226820	1716	1716	R	90.9	0.0	0:26.82	yes
	134	root	20	0	0	0	0	I	9.1	0.0	0:00.28	kworker/u17:3-events_unbound
	1	root	20	0	49192	41144	10256	S	0.0	1.0	0:02.09	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	0:00.06	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
	16	root	20	0	0	0	0	I	0.0	0.0	0:00.00	rcu_tasks_trace_kthread
	17	root	20	0	0	0	0	S	0.0	0.0	0:00.01	ksoftirqd/0
	18	root	20	0	0	0	0	I	0.0	0.0	0:00.16	rcu_preempt
	19	root	20	0	0	0	0	S	0.0	0.0	0:00.00	rcu_exp_par_gp_kthread_worker/0
	20	root	20	0	0	0	0	S	0.0	0.0	0:00.12	rcu_exp_gp_kthread_worker
	21	root	rt	0	0	0	0	S	0.0	0.0	0:00.11	migration/0

Рис. 3.11: Вывод top с процессами yes

9. Запущено ещё несколько процессов **yes** в фоновом режиме. Для завершения части из них использованы команды:

- **kill <PID>** — завершение по PID
- **kill %<номер>** — завершение по идентификатору задания

```

root@haoladar:/home/haoladar# yes > /dev/null &
[1] 5839
root@haoladar:/home/haoladar# yes > /dev/null &
[2] 5841
root@haoladar:/home/haoladar# yes > /dev/null &
[3] 5843
root@haoladar:/home/haoladar# kill 5839
root@haoladar:/home/haoladar#
[1] Terminated yes > /dev/null
root@haoladar:/home/haoladar# fg 2
yes > /dev/null
^C
root@haoladar:/home/haoladar#
root@haoladar:/home/haoladar# kill -1 5324
root@haoladar:/home/haoladar# kill -1 5843
[3]+ Hangup yes > /dev/null
root@haoladar:/home/haoladar#
root@haoladar:/home/haoladar# jobs
root@haoladar:/home/haoladar# yes > /dev/null &
[1] 6015
root@haoladar:/home/haoladar# yes > /dev/null &
[2] 6017
root@haoladar:/home/haoladar# yes > /dev/null &
[3] 6019
root@haoladar:/home/haoladar# killall yes
[1] Terminated yes > /dev/null
[2]- Terminated yes > /dev/null
[3]+ Terminated yes > /dev/null
root@haoladar:/home/haoladar# █

```

Рис. 3.12: Завершение процессов по PID и job ID

10. Для проверки обработки сигналов был отправлен сигнал **SIGHUP (1)** обычному процессу и процессу, запущенному через **nohup**.

Первый завершился, второй продолжил работу.

11. Запущено несколько новых процессов **yes**, которые затем были завершены одной командой:

killall yes

12. Для сравнения приоритетов два процесса были запущены так:

- **yes > /dev/null &** (обычный приоритет)
- **nice -n 5 yes > /dev/null &** (с приоритетом +5).

Проверка через **ps -l** показала различие в приоритетах (*NI*).


```

root@haoladar:/home/haoladar# yes > /dev/null &
[1] 6200
root@haoladar:/home/haoladar# nice -n 5 yes > /dev/null &
[2] 6223
root@haoladar:/home/haoladar# ps -l
F S  UID      PID      PPID  C PRI  NI ADDR SZ WCHAN  TTY          TIME CMD
4 S   0        3319      3275  0  80   0 - 58153 do_wai pts/0      00:00:00 su
4 S   0        3351      3319  0  80   0 - 57575 do_wai pts/0      00:00:00 bash
4 R   0        6200      3351 99  80   0 - 56705 -      pts/0      00:00:11 yes
4 R   0        6223      3351 99  85   5 - 56705 -      pts/0      00:00:04 yes
4 R   0        6235      3351  0  80   0 - 57682 -      pts/0      00:00:00 ps
root@haoladar:/home/haoladar# renice -n 5 6200
6200 (process ID) old priority 0, new priority 5
root@haoladar:/home/haoladar# ps -l
F S  UID      PID      PPID  C PRI  NI ADDR SZ WCHAN  TTY          TIME CMD
4 S   0        3319      3275  0  80   0 - 58153 do_wai pts/0      00:00:00 su
4 S   0        3351      3319  0  80   0 - 57575 do_wai pts/0      00:00:00 bash
4 R   0        6200      3351 99  85   5 - 56705 -      pts/0      00:00:25 yes
4 R   0        6223      3351 99  85   5 - 56705 -      pts/0      00:00:18 yes
4 R   0        6260      3351  0  80   0 - 57682 -      pts/0      00:00:00 ps
root@haoladar:/home/haoladar# killall yes
[1]-  Terminated                  yes > /dev/null
[2]+  Terminated                  nice -n 5 yes > /dev/null
root@haoladar:/home/haoladar#

```

Рис. 3.13: Сравнение приоритетов процессов

13. Один из процессов был изменён с помощью **renice**, чтобы выровнять приоритеты обоих потоков.

```

root@haoladar:/home/haoladar# yes > /dev/null &
[1] 6200
root@haoladar:/home/haoladar# nice -n 5 yes > /dev/null &
[2] 6223
root@haoladar:/home/haoladar# ps -l
F S  UID      PID      PPID  C PRI  NI ADDR SZ WCHAN  TTY          TIME CMD
4 S   0        3319      3275  0  80   0 - 58153 do_wai pts/0      00:00:00 su
4 S   0        3351      3319  0  80   0 - 57575 do_wai pts/0      00:00:00 bash
4 R   0        6200      3351 99  80   0 - 56705 -      pts/0      00:00:11 yes
4 R   0        6223      3351 99  85   5 - 56705 -      pts/0      00:00:04 yes
4 R   0        6235      3351  0  80   0 - 57682 -      pts/0      00:00:00 ps
root@haoladar:/home/haoladar# renice -n 5 6200
6200 (process ID) old priority 0, new priority 5
root@haoladar:/home/haoladar# ps -l
F S  UID      PID      PPID  C PRI  NI ADDR SZ WCHAN  TTY          TIME CMD
4 S   0        3319      3275  0  80   0 - 58153 do_wai pts/0      00:00:00 su
4 S   0        3351      3319  0  80   0 - 57575 do_wai pts/0      00:00:00 bash
4 R   0        6200      3351 99  85   5 - 56705 -      pts/0      00:00:25 yes
4 R   0        6223      3351 99  85   5 - 56705 -      pts/0      00:00:18 yes
4 R   0        6260      3351  0  80   0 - 57682 -      pts/0      00:00:00 ps
root@haoladar:/home/haoladar# killall yes
[1]-  Terminated                  yes > /dev/null
[2]+  Terminated                  nice -n 5 yes > /dev/null
root@haoladar:/home/haoladar#

```

Рис. 3.14: Изменение приоритетов с помощью renice

4 Контрольные вопросы

1. Какая команда позволяет вам искать пакет `rpm`, содержащий файл `useradd`?
`rpm -qf /usr/sbin/useradd` — если файл уже установлен.
`dnf provides /useradd` или `repoquery -f /useradd` — если нужно найти пакет, в который входит этот файл.
2. Какие команды вам нужно использовать, чтобы показать имя группы `dnf`, которая содержит инструменты безопасности и показать, что находится в этой группе?
 - **`dnf group list`** — список всех групп.
 - **`dnf group info "Security Tools"`** — подробности о содержимом группы.
3. Какая команда позволяет вам установить `rpm`, который вы загрузили из Интернета и который не находится в репозиториях?
`rpm -Uhv имя_пакета.rpm`
(или с проверкой зависимостей: **`dnf install ./имя_пакета.rpm`**).
4. Вы хотите убедиться, что пакет `rpm`, который вы загрузили, не содержит никакого опасного кода сценария. Какая команда позволяет это сделать?
`rpm -qp --scripts имя_пакета.rpm` — показывает скрипты, которые выполняются при установке/удалении.
5. Какая команда показывает всю документацию в `rpm`?
`rpm -qd имя_пакета`

6. Какая команда показывает, какому пакету grp принадлежит файл?

grp -qf /путь/к/файлу

5 Заключение

Освоены базовые навыки работы с пакетным менеджером **dnf** и утилитой **rpm** для установки, поиска, изучения и удаления пакетов и групп в Linux.