

Отчёт по лабораторной работе №6

Управление процессами

Анастасия Мазуркевич

Содержание

1	Цель работы	5
2	Ход выполнения	6
2.1	Управление заданиями	6
2.2	Управление процессами	9
2.3	Задание 1	10
2.4	Задание 2	11
3	Контрольные вопросы	16
4	Заключение	18

Список иллюстраций

2.1	Список запущенных заданий	7
2.2	Процесс dd в top	8
2.3	Завершение процесса dd в top	8
2.4	Поиск процессов dd	9
2.5	Завершение родительского процесса и дочерних dd	10
2.6	Запуск, изменение приоритета и завершение процессов dd	11
2.7	Фоновые и приостановленные процессы yes	12
2.8	Запуск yes с nohup	12
2.9	Процессы yes в top	13
2.10	Завершение процессов yes	14
2.11	Изменение приоритета процессов yes	15

Список таблиц

1 Цель работы

Получить навыки управления процессами операционной системы.

2 Ход выполнения

2.1 Управление заданиями

Сначала были получены полномочия администратора с помощью команды:

```
su -
```

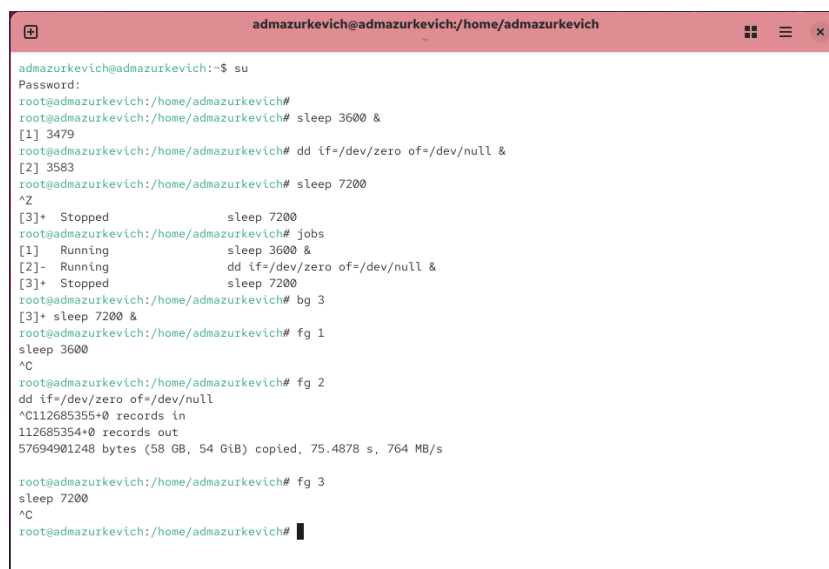
Далее последовательно запущены процессы:

```
sleep 3600 & dd if=/dev/zero of=/dev/null & sleep 7200
```

Поскольку последняя команда была выполнена без «&», терминал заблокировался на 2 часа. Чтобы вернуть управление оболочкой, использовалось сочетание клавиш Ctrl+Z, после чего процесс был приостановлен.

Командой jobs просмотрен список фоновых заданий:

- sleep 3600 и dd if=/dev/zero of=/dev/null находились в состоянии Running,
- sleep 7200 был отмечен как Stopped.



```
admazurkevich@admazurkevich:/home/admazurkevich
admazurkevich@admazurkevich:~$ su
Password:
root@admazurkevich:/home/admazurkevich#
root@admazurkevich:/home/admazurkevich# sleep 3600 &
[1] 3479
root@admazurkevich:/home/admazurkevich# dd if=/dev/zero of=/dev/null &
[2] 3583
root@admazurkevich:/home/admazurkevich# sleep 7200
^Z
[3]+  Stopped                  sleep 7200
root@admazurkevich:/home/admazurkevich# jobs
[1]  Running                  sleep 3600 &
[2]-  Running                  dd if=/dev/zero of=/dev/null &
[3]+  Stopped                  sleep 7200
root@admazurkevich:/home/admazurkevich# bg 3
[3]+ sleep 7200 &
root@admazurkevich:/home/admazurkevich# fg 1
sleep 3600
^C
root@admazurkevich:/home/admazurkevich# fg 2
dd if=/dev/zero of=/dev/null
^C112685355+0 records in
112685354+0 records out
57694901248 bytes (58 GB, 54 GiB) copied, 75.4878 s, 764 MB/s
root@admazurkevich:/home/admazurkevich# fg 3
sleep 7200
^C
root@admazurkevich:/home/admazurkevich#
```

Рис. 2.1: Список запущенных заданий

Для возобновления выполнения третьего задания в фоне применена команда:

bg 3

Затем команда **fg 1** переместила процесс **sleep 3600** на передний план.

После нажатия **Ctrl+C** задание завершилось. Аналогичные действия выполнены для оставшихся процессов (**dd** и **sleep 7200**).

В другом терминале, под обычным пользователем, было запущено:

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

После выхода из терминала командой **exit** процесс продолжил работу в системе.

Для проверки активности процессов использовалась команда **top**. На экране отобразилось, что процесс **dd** выполняется, занимая более 90% CPU.

```
top - 15:06:38 up 6 min, 4 users, load average: 0.47, 0.34, 0.17
Tasks: 260 total, 2 running, 258 sleeping, 0 stopped, 0 zombie
%Cpu(s): 11.6 us, 16.3 sy, 0.0 ni, 72.1 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st
MiB Mem : 3909.0 total, 1308.4 free, 1419.7 used, 1419.4 buff/cache
MiB Swap: 4040.0 total, 4040.0 free, 0.0 used, 2489.3 avail Mem
```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
3905	admazur+	20	0	226848	1808	1808	R	90.9	0.0	0:24.67	dd
1	root	20	0	49192	41140	10216	S	0.0	1.0	0:01.15	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
8	root	20	0	0	0	0	I	0.0	0.0	0:00.02	kworker/0:0-events
10	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/0:0H-xfs-log/dm-0
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	0:00.00	ksoftirqd/0
18	root	20	0	0	0	0	I	0.0	0.0	0:00.04	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	0:00.01	rcu_exp_gp_kthread_worker
21	root	rt	0	0	0	0	S	0.0	0.0	0:00.00	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
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	0:00.14	migration/1
27	root	20	0	0	0	0	S	0.0	0.0	0:00.00	ksoftirqd/1

Рис. 2.2: Процесс dd в top

После выхода из top клавишей q программа была запущена повторно. С помощью клавиши k процесс dd был завершён. Это подтвердилось исчезновением задания из списка.

```
top - 15:07:16 up 7 min, 4 users, load average: 0.73, 0.43, 0.20
Tasks: 260 total, 2 running, 258 sleeping, 0 stopped, 0 zombie
%Cpu(s): 5.8 us, 8.1 sy, 0.1 ni, 85.8 id, 0.0 wa, 0.2 hi, 0.0 si, 0.0 st
MiB Mem : 3909.0 total, 1261.9 free, 1465.7 used, 1420.0 buff/cache
MiB Swap: 4040.0 total, 4040.0 free, 0.0 used, 2443.3 avail Mem
```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
3241	admazur+	20	0	3031184	324704	96656	S	3.5	8.1	0:03.01	ptxkis
2150	admazur+	20	0	4850896	311748	123120	R	2.2	7.8	0:03.98	gnome-shell
707	root	20	0	0	0	0	I	0.8	0.0	0:00.20	kworker/u17:4-events_unbound
1	root	20	0	49192	41140	10216	S	0.0	1.0	0:01.17	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
8	root	20	0	0	0	0	I	0.0	0.0	0:00.02	kworker/0:0-events
10	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/0:0H-xfs-log/dm-0
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
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	0:00.00	ksoftirqd/0
18	root	20	0	0	0	0	I	0.0	0.0	0:00.04	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	0:00.01	rcu_exp_gp_kthread_worker
21	root	rt	0	0	0	0	S	0.0	0.0	0:00.00	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
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

Рис. 2.3: Завершение процесса dd в top

2.2 Управление процессами

Сначала были получены полномочия администратора с помощью команды:

```
su -
```

Затем запущены три процесса:

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

Для просмотра списка процессов использовалась команда:

```
ps aux | grep dd
```

Она показала все строки, содержащие dd. Внизу списка отобразились три активных процесса dd.

```
root@admazurkevich:/home/admazurkevich# dd if=/dev/zero of=/dev/zero &  
[1] 4261  
root@admazurkevich:/home/admazurkevich# dd if=/dev/zero of=/dev/zero &  
[2] 4263  
root@admazurkevich:/home/admazurkevich# dd if=/dev/zero of=/dev/zero &  
[3] 4265  
root@admazurkevich:/home/admazurkevich# ps aux | grep dd  
root      2  0.0  0.0   0   0 ?        S   14:59   0:00 [kthreadd]  
root     92  0.0  0.0   0   0 ?        I<  14:59   0:00 [kworker/R-ipv6_addrconf]  
root    1155  0.0  0.0 578492 3140 ?        SL  14:59   0:00 /usr/sbin/VBoxService --pidfile /var/run/vboxadd-s  
ervice.sh  
admazur+ 2525  0.0  0.6 962676 25636 ?        Ssl 15:01   0:00 /usr/libexec/evolution-addressbook-factory  
root     4261 99.5  0.0 226848 1760 pts/0    R   15:08   0:12 dd if=/dev/zero of=/dev/zero  
root     4263 98.7  0.0 226848 1752 pts/0    R   15:08   0:10 dd if=/dev/zero of=/dev/zero  
root     4265 99.2  0.0 226848 1816 pts/0    R   15:08   0:10 dd if=/dev/zero of=/dev/zero  
root     4291  0.0  0.0 227688 2028 pts/0    S+  15:08   0:00 grep --color=auto dd  
root@admazurkevich:/home/admazurkevich# renice -n 5 4261  
4261 (process ID) old priority 0, new priority 5  
root@admazurkevich:/home/admazurkevich# ps fax | grep -B5 dd  
PID TTY      STAT     TIME COMMAND  
  2 ?        S         0:00 [kthreadd]  
--  
 82 ?        I<         0:00 \. [kworker/R-kthrotld]  
 87 ?        I<         0:00 \. [kworker/R-acpi_thermal_pm]  
 88 ?        I<         0:00 \. [kworker/R-kmpath_rdacd]  
 89 ?        I<         0:00 \. [kworker/R-kalud]  
 91 ?        I<         0:00 \. [kworker/R-mld]  
 92 ?        I<         0:00 \. [kworker/R-ipv6_addrconf]  
--  
 928 ?      SNs       0:00 /usr/sbin/alsactl -s -n 19 -c -E ALSA_CONFIG_PATH=/etc/alsa/alsactl.conf --initfile=/lib  
/alsa/intel00main daemon  
053 ?      S         0:00 /usr/sbin/chronpd -E ?
```

Рис. 2.4: Поиск процессов dd

Далее для одного из процессов был изменён приоритет командой:

```
renice -n 5 <PID>
```

После этого для наглядного просмотра иерархии процессов применялась ко-
манда:

```
ps fax | grep -B5 dd
```

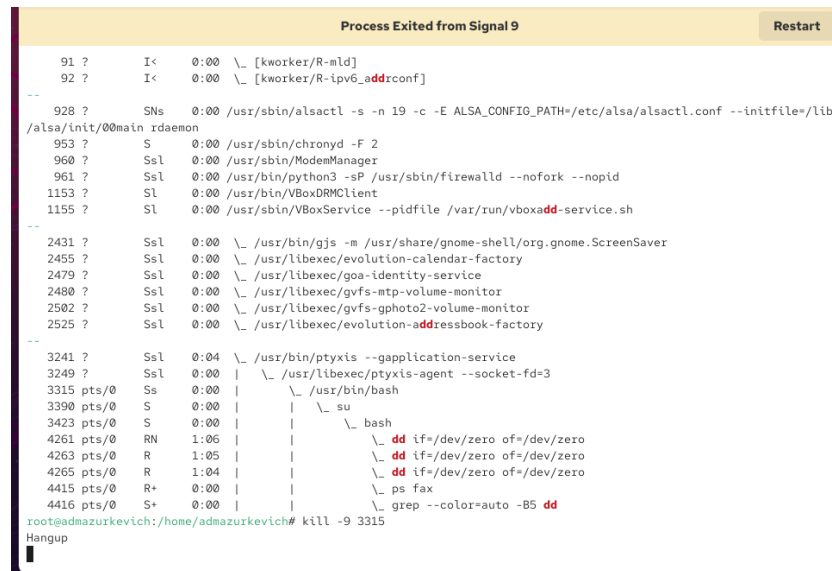
Опция -B5 позволила вывести несколько строк выше совпадения, что дало воз-
можность увидеть родительскую оболочку, из которой были запущены процессы

dd, вместе с её PID.

Затем был найден PID оболочки, и выполнена команда:

```
kill -9 <PID>
```

После завершения родительской оболочки автоматически остановились и все дочерние процессы dd.



```
Process Exited from Signal 9 Restart
91 ? I< 0:00 \_ [kworker/R-mld]
92 ? I< 0:00 \_ [kworker/R-ipv6_dddconf]
--
928 ? Sns 0:00 /usr/sbin/alsactl -s -n 19 -c -E ALSA_CONFIG_PATH=/etc/alsa/alsactl.conf --initfile=/lib
/alsa/init/00main rdaemon
953 ? S 0:00 /usr/sbin/chronyd -F 2
960 ? Ssl 0:00 /usr/sbin/ModemManager
961 ? Ssl 0:00 /usr/bin/python3 -sP /usr/sbin/firewalld --nofork --nopid
1153 ? Sl 0:00 /usr/bin/VBoxDRMClient
1155 ? Sl 0:00 /usr/sbin/VBoxService --pidfile /var/run/vboxadd-service.sh
--
2431 ? Ssl 0:00 \_ /usr/bin/gjs -m /usr/share/gnome-shell/org.gnome.ScreenSaver
2455 ? Ssl 0:00 \_ /usr/libexec/evolution-calendar-factory
2479 ? Ssl 0:00 \_ /usr/libexec/goa-identity-service
2480 ? Ssl 0:00 \_ /usr/libexec/gvfs-mtp-volume-monitor
2502 ? Ssl 0:00 \_ /usr/libexec/gvfs-gphoto2-volume-monitor
2525 ? Ssl 0:00 \_ /usr/libexec/evolution-addrbook-factory
--
3241 ? Ssl 0:04 \_ /usr/bin/ptxix --gaplication-service
3249 ? Ssl 0:00 | \_ /usr/libexec/ptxix-agent --socket-fd=3
3315 pts/0 Ss 0:00 | \_ /usr/bin/bash
3390 pts/0 S 0:00 | | \_ su
3423 pts/0 S 0:00 | | \_ bash
4261 pts/0 RN 1:06 | | \_ dd if=/dev/zero of=/dev/zero
4263 pts/0 R 1:05 | | \_ dd if=/dev/zero of=/dev/zero
4265 pts/0 R 1:04 | | \_ dd if=/dev/zero of=/dev/zero
4415 pts/0 R+ 0:00 | | \_ ps fax
4416 pts/0 S+ 0:00 | | \_ grep --color=auto -B5 dd
root@admazurkevich:/home/admazurkevich# kill -9 3315
Hangup
```

Рис. 2.5: Завершение родительского процесса и дочерних dd

2.3 Задание 1

Сначала были получены права администратора командой:

```
su
```

Затем трижды был запущен процесс:

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

Каждый процесс был помещён в фоновый режим и получил собственный PID.

Для одного из процессов был изменён приоритет с помощью команды:

```
renice -n 5 <PID>
```

Вывод показал: старый приоритет 0, новый приоритет 5.

После этого для того же процесса приоритет был изменён снова:

```
renice -n 15 <PID>
```

Теперь в выводе указано: старый приоритет 5, новый приоритет 15.

В завершение все запущенные процессы dd были остановлены с помощью команды:

```
killall dd
```

Результат показал, что все три процесса завершены.

```
admazurkevich@admazurkevich:~$ su
Password:
root@admazurkevich:/home/admazurkevich# dd if=/dev/zero of=/dev/null &
[1] 4885
root@admazurkevich:/home/admazurkevich# dd if=/dev/zero of=/dev/null &
[2] 4887
root@admazurkevich:/home/admazurkevich# dd if=/dev/zero of=/dev/null &
[3] 4889
root@admazurkevich:/home/admazurkevich# renice -n 5 4885
4885 (process ID) old priority 0, new priority 5
root@admazurkevich:/home/admazurkevich# renice -n 15 4885
4885 (process ID) old priority 5, new priority 15
root@admazurkevich:/home/admazurkevich# killall dd
[2]- Terminated          dd if=/dev/zero of=/dev/null
[1]- Terminated          dd if=/dev/zero of=/dev/null
[3]+ Terminated          dd if=/dev/zero of=/dev/null
root@admazurkevich:/home/admazurkevich#
```

Рис. 2.6: Запуск, изменение приоритета и завершение процессов dd

2.4 Задание 2

Сначала была запущена программа:

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

Процесс был помещён в фоновый режим. Затем аналогичная команда выполнена на переднем плане и приостановлена сочетанием Ctrl+Z. После этого процесс был завершён с помощью Ctrl+C.

Для проверки статуса заданий использовалась команда:

```
jobs
```

Вывод показал, что один процесс находился в состоянии Running, другой — Stopped.

```

root@admazurkevich:/home/admazurkevich# yes > /dev/null &
[1] 5097
root@admazurkevich:/home/admazurkevich# yes > /dev/null
^Z
[2]+  Stopped                  yes > /dev/null
root@admazurkevich:/home/admazurkevich# yes > /dev/null
^C
root@admazurkevich:/home/admazurkevich#
root@admazurkevich:/home/admazurkevich# jobs
[1]-  Running                  yes > /dev/null &
[2]+  Stopped                  yes > /dev/null
root@admazurkevich:/home/admazurkevich#

```

Рис. 2.7: Фоновые и приостановленные процессы yes

Следующим шагом один из процессов был перемещён на передний план командой `fg 1` и остановлен. Другой процесс был возобновлён в фоне командой `bg 2`. Для проверки снова применялась команда `jobs`, где оба процесса отмечены как `Running`.

Далее процесс был запущен с опцией `nohup`:

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

Благодаря этому он продолжал выполняться даже после выхода из оболочки.

```

root@admazurkevich:/home/admazurkevich#
root@admazurkevich:/home/admazurkevich# jobs
[1]-  Running                  yes > /dev/null &
[2]+  Stopped                  yes > /dev/null
root@admazurkevich:/home/admazurkevich# fg 1
yes > /dev/null
^C
root@admazurkevich:/home/admazurkevich#
root@admazurkevich:/home/admazurkevich# jobs
[2]+  Stopped                  yes > /dev/null
root@admazurkevich:/home/admazurkevich# bg 2
[2]+ yes > /dev/null &
root@admazurkevich:/home/admazurkevich# jobs
[2]+  Running                  yes > /dev/null &
root@admazurkevich:/home/admazurkevich# nohup yes > /dev/null &
[3] 5275
nohup: ignoring input and redirecting stderr to stdout
root@admazurkevich:/home/admazurkevich# jobs
[2]-  Running                  yes > /dev/null &
[3]+  Running                  nohup yes > /dev/null &
root@admazurkevich:/home/admazurkevich#

```

Рис. 2.8: Запуск yes с nohup

Затем была запущена утилита `top`. В её выводе отобразились два процесса `yes`, оба активно использовали процессорное время.

```

top - 15:15:49 up 15 min, 5 users, load average: 1.27, 1.57, 0.95
Tasks: 262 total, 3 running, 259 sleeping, 0 stopped, 0 zombie
%Cpu(s): 7.7 us, 19.2 sy, 0.0 ni, 73.1 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st
MiB Mem : 3909.0 total, 1334.3 free, 1356.3 used, 1457.5 buff/cache
MiB Swap: 4040.0 total, 4040.0 free, 0.0 used, 2552.7 avail Mem

```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
5109	root	20	0	226820	1760	1760	R	100.0	0.0	0:36.17	yes
5275	root	20	0	226820	1716	1716	R	100.0	0.0	0:19.27	yes
1	root	20	0	49192	41140	10216	S	0.0	1.0	0:01.62	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
8	root	20	0	0	0	0	I	0.0	0.0	0:00.02	kworker/0:0-mm_percpu_wq
10	root	0	-20	0	0	0	I	0.0	0.0	0:00.01	kworker/0:0H-xfs-log/dm-0
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	0:00.05	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	0:00.00	ksoftirqd/0
18	root	20	0	0	0	0	I	0.0	0.0	0:00.12	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	0:00.00	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
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	0:00.14	migration/1

Рис. 2.9: Процессы yes в top

Были запущены ещё несколько процессов yes. Для их завершения использовались разные способы:

- по PID с помощью команды kill,
- по идентификатору задания через fg и остановку,
- а также массово через команду killall yes.

```

root@admazurkevich:/home/admazurkevich# yes > /dev/null &
[1] 5566
root@admazurkevich:/home/admazurkevich# yes > /dev/null &
[2] 5571
root@admazurkevich:/home/admazurkevich# yes > /dev/null &
[3] 5583
root@admazurkevich:/home/admazurkevich# kill 5583
[3]+  Terminated          yes > /dev/null
root@admazurkevich:/home/admazurkevich#
root@admazurkevich:/home/admazurkevich# fg 2
yes > /dev/null
^C
root@admazurkevich:/home/admazurkevich#
root@admazurkevich:/home/admazurkevich# kill -1 5566
[1]+  Hangup                yes > /dev/null
root@admazurkevich:/home/admazurkevich# kill -1 5109
root@admazurkevich:/home/admazurkevich#
root@admazurkevich:/home/admazurkevich# kill -1 5275
root@admazurkevich:/home/admazurkevich#
root@admazurkevich:/home/admazurkevich# yes > /dev/null &
[1] 5844
root@admazurkevich:/home/admazurkevich# yes > /dev/null &
[2] 5846
root@admazurkevich:/home/admazurkevich# yes > /dev/null &
[3] 5848
root@admazurkevich:/home/admazurkevich# yes > /dev/null &
[4] 5850
root@admazurkevich:/home/admazurkevich# killall yes
[4]+  Terminated          yes > /dev/null
[1]  Terminated          yes > /dev/null
[2]-  Terminated          yes > /dev/null
[3]+  Terminated          yes > /dev/null
root@admazurkevich:/home/admazurkevich#
root@admazurkevich:/home/admazurkevich#

```

Рис. 2.10: Завершение процессов yes

В завершение одно из заданий yes было запущено с изменённым приоритетом:

```
nice -n 5 yes > /dev/null &
```

С помощью команды ps -l было показано, что у данного процесса приоритет увеличился (число nice стало больше). Затем утилитой renice приоритет был изменён у другого процесса так, чтобы значения у них совпадали.

```

root@admazurkevich:/home/admazurkevich# yes > /dev/null &
[1] 5980
root@admazurkevich:/home/admazurkevich# nice -n 5 yes > /dev/null &
[2] 6004
root@admazurkevich:/home/admazurkevich# ps -l
F S  UID      PID  PPID  C PRI  NI ADDR SZ WCHAN  TTY          TIME CMD
4 S   0       5317   3954  0  80   0 - 58153 do_wai pts/2    00:00:00 su
4 S   0       5343   5317  0  80   0 - 57575 do_wai pts/2    00:00:00 bash
4 R   0       5980   5343  99  80   0 - 56705 -      pts/2    00:00:09 yes
4 R   0       6004   5343  98  85   5 - 56705 -      pts/2    00:00:02 yes
4 R   0       6006   5343  0  80   0 - 57682 -      pts/2    00:00:00 ps
root@admazurkevich:/home/admazurkevich# renice -n 5 5980
5980 (process ID) old priority 0, new priority 5
root@admazurkevich:/home/admazurkevich# ps -l
F S  UID      PID  PPID  C PRI  NI ADDR SZ WCHAN  TTY          TIME CMD
4 S   0       5317   3954  0  80   0 - 58153 do_wai pts/2    00:00:00 su
4 S   0       5343   5317  0  80   0 - 57575 do_wai pts/2    00:00:00 bash
4 R   0       5980   5343  99  85   5 - 56705 -      pts/2    00:00:54 yes
4 R   0       6004   5343  99  85   5 - 56705 -      pts/2    00:00:47 yes
4 R   0       6103   5343  0  80   0 - 57682 -      pts/2    00:00:00 ps
root@admazurkevich:/home/admazurkevich# killall yes
[1]-  Terminated                  yes > /dev/null
[2]+  Terminated                  nice -n 5 yes > /dev/null
root@admazurkevich:/home/admazurkevich#

```

Рис. 2.11: Изменение приоритета процессов yes

3 Контрольные вопросы

1. Какая команда даёт обзор всех текущих заданий оболочки?

Команда `jobs`. Она выводит список процессов, запущенных из текущей оболочки, с их статусом (Running, Stopped и др.).

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

Сначала приостановить процесс сочетанием клавиш `Ctrl+Z`, затем возобновить его в фоне командой `bg`.

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

`Ctrl+C` — отправляет сигнал прерывания (SIGINT), завершающий процесс.

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 <PID>` или `killall <имя_процесса>` из другой оболочки или от имени администратора.

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

`ps fax` — выводит дерево процессов с отображением иерархии.

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

`renice -n -5 -p 1234` — уменьшение значения `nice` повышает приоритет процесса.

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

Командой `killall dd`, которая завершит все процессы с указанным именем.

8. Какая команда позволяет остановить команду с именем mycommand?

`killall mycommand`.

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

Внутри `top` используется клавиша `k`, после чего вводится PID процесса для завершения.

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

С помощью `nice` с положительным значением, например:

`nice -n 10 <команда>` — это понижает приоритет задачи, снижая нагрузку на систему.

4 Заключение

В процессе лабораторной работы были освоены основные приёмы администрирования пользователей и групп в Linux: создание и настройка учётных записей, редактирование параметров паролей, работа с `sudo` и назначение групповых прав. Эти навыки формируют базу для эффективного управления системой.