

Отчёт по лабораторной работе №6

Артём Дмитриевич Петлин

Содержание

1	Цель работы	5
2	Задание	6
3	Теоретическое введение	7
4	Выполнение лабораторной работы	8
5	Выводы	20
	Список литературы	21

Список иллюстраций

4.1	su -	8
4.2	bg fg	9
4.3	ctrl + z	9
4.4	&	10
4.5	top	10
4.6	top	10
4.7	su -	11
4.8	renice	11
4.9	ps fax grep -B5 dd	12
4.10	kill -9	12
4.11	renice	13
4.12	killall -9	13
4.13	yes	14
4.14	ctrl + z	14
4.15	ctrl + c	14
4.16	yes	15
4.17	yes	15
4.18	nohup	16
4.19	ps -A	16
4.20	yes	17
4.21	kill -1	17
4.22	killall	18
4.23	yes	19
4.24	renice	19

Список таблиц

1 Цель работы

Получить навыки управления процессами операционной системы.

2 Задание

1. Продемонстрируйте навыки управления заданиями операционной системы (см. раздел 6.4.1).
2. Продемонстрируйте навыки управления процессами операционной системы (см. раздел 6.4.2).
3. Выполните задания для самостоятельной работы (см. раздел 6.5)

3 Теоретическое введение

Под процессом в операционной системе понимается абстракция, описывающая выполняющуюся программу. Информацию о выполняющихся в операционной системе типа Unix процессах можно получить, например, с помощью команд `ps`, `top`, `htop`.

4 Выполнение лабораторной работы

```
adpetlin@adpetlin:~$ su -
Password:
Last login: Fri Oct  3 10:22:04 MSK 2025 on tty1
root@adpetlin:~# sleep 3600 &
[1] 4294
root@adpetlin:~# dd if=/dev/zero of=/dev/null &
[2] 4311
root@adpetlin:~# sleep 7200
^Z
[3]+  Stopped                  sleep 7200
root@adpetlin:~# jobs
[1]  Running                  sleep 3600 &
[2]-  Running                  dd if=/dev/zero of=/dev/null &
[3]+  Stopped                  sleep 7200
root@adpetlin:~# █
```

Рисунок 4.1: su -

Получаем полномочия администратора. Запускаем несколько фоновых процессов и один длительный процесс на переднем плане. Введите Ctrl + z , чтобы остановить процесс.


```

root@adpetlin:~# bg 3
[3]+ sleep 7200 &
root@adpetlin:~# jobs
[1]  Running                sleep 3600 &
[2]-  Running                dd if=/dev/zero of=/dev/null &
[3]+  Running                sleep 7200 &
root@adpetlin:~# fg 1
sleep 3600
^C
root@adpetlin:~# jobs
[2]-  Running                dd if=/dev/zero of=/dev/null &
[3]+  Running                sleep 7200 &
root@adpetlin:~# █

```

Рисунок 4.2: bg | fg

Возобновляем выполнение приостановленного задания в фоновом режиме и наблюдаем изменение его статуса. Перемещаем одно из заданий на передний план. Завершаем выполнение задания на переднем плане с помощью комбинации клавиш и проверяем изменения в списке заданий.

```

root@adpetlin:~# fg 2
dd if=/dev/zero of=/dev/null
^C216327479+0 records in
216327478+0 records out
110759668736 bytes (111 GB, 103 GiB) copied, 122.092 s, 907 MB/s

root@adpetlin:~# fg 3
sleep 7200
^C
root@adpetlin:~# jobs
root@adpetlin:~# █

```

Рисунок 4.3: ctrl + z

Последовательно завершаем оставшиеся задания.

```
adpetlin@adpetlin:~$ dd if=/dev/zero of=/dev/null &
[1] 4497
adpetlin@adpetlin:~$ exir
bash: exir: command not found...
adpetlin@adpetlin:~$ exit
```

Рисунок 4.4: &

На втором терминале запускаем фоновый процесс от имени обычного пользователя. Закрываем второй терминал. На другом терминале проверяем, что процесс продолжает выполняться, с помощью системного монитора.

MiB Swap: 5120.0 total, 5120.0 free, 0.0 used. 3080.1 avail Mem											
PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
4497	adpetlin	20	0	226848	1828	1828	R	99.0	0.0	0:41.42	dd
1445	root	20	0	1129804	314688	24076	S	9.0	5.4	0:22.45	packageki

Рисунок 4.5: top

MiB Swap: 5120.0 total, 5120.0 free, 0.0 used. 2956.5 avail Mem											
PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
3962	adpetlin	20	0	4197632	287480	101224	S	7.3	5.0	0:11.07	ptyxis
2579	adpetlin	20	0	5574080	410604	137764	S	6.6	7.1	1:16.79	gnome-she

Рисунок 4.6: top

Завершаем оставшийся процесс через системный монитор.

```
adpetlin@adpetlin:~$ su -
Password:
Last login: Fri Oct 10 20:36:13 MSK 2025 on pts/0
root@adpetlin:~# dd if=/dev/zero of=/dev/null &
[1] 5018
root@adpetlin:~# dd if=/dev/zero of=/dev/null &
[2] 5019
root@adpetlin:~# dd if=/dev/zero of=/dev/null &
[3] 5025
root@adpetlin:~# ps aux | grep dd
root      2  0.0  0.0   0   0 ?        S   20:34   0:00 [kthreadd]
root     12  0.0  0.0   0   0 ?        I   20:34   0:00 [kworker/u24:1-ipv6_addrconf]
root    114  0.0  0.0   0   0 ?        I<  20:34   0:00 [kworker/R-ipv6_addrconf]
dbus     919  0.0  0.0  8200 5780 ?        S   20:34   0:00 dbus-broker --log 4 --controller 9 --machine-id e8
e6acddfb1248ed931b45f2e426c7b2 --max-bytes 536870912 --max-fds 4096 --max-matches 131072 --audit
root    1192  0.0  0.0 512956 2960 ?        Sl  20:34   0:00 /usr/sbin/VBoxService --pidfile /var/run/vboxadd-s
ervice.sh
adpetlin 2496  0.0  0.0  6716 4212 ?        S   20:34   0:00 dbus-broker --log 4 --controller 9 --machine-id e8
e6acddfb1248ed931b45f2e426c7b2 --max-bytes 1000000000000000 --max-fds 250000000000000 --max-matches 5000000000
adpetlin 2638  0.0  0.0  4756 2636 ?        S   20:34   0:00 dbus-broker --log 4 --controller 9 --machine-id e8
e6acddfb1248ed931b45f2e426c7b2 --max-bytes 1000000000000000 --max-fds 6400000 --max-matches 5000000000
adpetlin 2941  0.0  0.4 1036428 25176 ?      Ssl  20:34   0:00 /usr/libexec/evolution-addressbook-factory
adpetlin 3534  0.0  0.5 260704 32792 ?        Sl  20:35   0:00 /usr/lib64/firefox/firefox -contentproc -parentBui
ldID 20250918202509 -prefsHandle 0:35268 -prefMapHandle 1:271206 -sandboxReporter 2 -chrootClient 3 -ipcHandle 4 -ini
tialChannelId {4e07e16c-d5cd-4d9a-a59e-13675fdcb99e} -parentPid 3402 -appDir /usr/lib64/firefox/browser 3 rdd
root     5018 98.8  0.0 226848 1896 pts/0    R   20:41   0:07 dd if=/dev/zero of=/dev/null
root     5019 98.7  0.0 226848 1828 pts/0    R   20:41   0:06 dd if=/dev/zero of=/dev/null
root     5025 98.5  0.0 226848 1664 pts/0    R   20:41   0:06 dd if=/dev/zero of=/dev/null
root     5034  0.0  0.0 227688 2076 pts/0    S+  20:41   0:00 grep --color=auto dd
root@adpetlin:~#
```

Рисунок 4.7: su -

Получаем полномочия администратора. Запускаем несколько фоновых процессов. Просматриваем информацию о запущенных процессах, фильтруя нужные нам процессы.

```
root@adpetlin:~# renice -n 5 5018
5018 (process ID) old priority 0, new priority 5
```

Рисунок 4.8: renice

Изменяем приоритет одного из процессов, используя его идентификатор.

```

4816 ?      Ssl    0:02  \_ /usr/bin/ptyxis --gaplication-service
4823 ?      Ssl    0:00      \_ /usr/libexec/ptyxis-agent --socket-fd=3
4880 pts/0   Ss     0:00          \_ /usr/bin/bash
4908 pts/0   S      0:00              \_ su -
4954 pts/0   S      0:00                  \_ -bash
5018 pts/0   RN     0:55                      \_ dd if=/dev/zero of=/dev/null
5019 pts/0   R      0:54                      \_ dd if=/dev/zero of=/dev/null
5025 pts/0   R      0:53                      \_ dd if=/dev/zero of=/dev/null
5078 pts/0   R+     0:00                      \_ ps fax
5079 pts/0   S+     0:00                      \_ grep --color=auto -B5 dd
oot@adpetlin:~#

```

Рисунок 4.9: ps fax | grep -B5 dd

Изучаем иерархию процессов, просматривая связи между родительскими и дочерними процессами.

```

270  kill -9 4816
271  history
root@adpetlin:~#

```

Рисунок 4.10: kill -9

Завершаем родительский процесс, что приводит к автоматическому завершению всех связанных дочерних процессов.

```

root@adpetlin:~# dd if=/dev/zero of=/dev/null &
[1] 5433
root@adpetlin:~# dd if=/dev/zero of=/dev/null &
[2] 5438
root@adpetlin:~# dd if=/dev/zero of=/dev/null &
[3] 5439
root@adpetlin:~# renice -n 5 5438
5438 (process ID) old priority 0, new priority 5
root@adpetlin:~# renice -n -5 5438
5438 (process ID) old priority 5, new priority -5
root@adpetlin:~# renice -n -15 5438
5438 (process ID) old priority -5, new priority -15
root@adpetlin:~# top

```

Рисунок 4.11: renice

Запускаем три фоновых процесса. Повышаем приоритет одного из процессов, устанавливая отрицательное значение. Дополнительно изменяем приоритет того же процесса, устанавливая другое значение, и анализируем разницу в приоритетах.

```

root@adpetlin:~# kill -9 5433
root@adpetlin:~# kill -9 5438
[1] Killed dd if=/dev/zero of=/dev/null
root@adpetlin:~# kill -9 5439
root@adpetlin:~# jobs
[2]- Killed dd if=/dev/zero of=/dev/null
[3]+ Killed dd if=/dev/zero of=/dev/null
root@adpetlin:~# fg 2
-bash: fg: 2: no such job
root@adpetlin:~# jobs
root@adpetlin:~#

```

Рисунок 4.12: kill -9

Завершаем все запущенные процессы.

```

root@adpetlin:~# yes > /dev/null &
[1] 6089
root@adpetlin:~# fg 1 > /dev/null
^Z
[1]+  Stopped                  yes > /dev/null
root@adpetlin:~# fg 1 > /dev/null
^Croot@adpetlin:~# █

```

Рисунок 4.13: yes

Запускаем фоновый процесс с перенаправлением вывода. Запускаем процесс на переднем плане с перенаправлением вывода, приостанавливаем его, затем возобновляем и завершаем.

```

y
y
y
y
y
y
^Z
[1]+  Stopped                  yes
root@adpetlin:~# fg 1 █

```

Рисунок 4.14: ctrl + z

```

y
y
y
y
y
y
y
y^C
root@adpetlin:~# █

```

Рисунок 4.15: ctrl + c

Запускаем процесс на переднем плане без перенаправления вывода, приостанавливаем, возобновляем и завершаем его.

```
root@adpetlin:~# jobs
root@adpetlin:~# yes > /dev/null &
[1] 6489
root@adpetlin:~# fg 1
yes > /dev/null
^C
root@adpetlin:~#
```

Рисунок 4.16: yes

Проверяем состояния всех заданий. Переводим фоновый процесс на передний план и останавливаем его.

```
root@adpetlin:~# yes > /dev/null
^Z
[1]+  Stopped                  yes > /dev/null
root@adpetlin:~# bg 1
[1]+ yes > /dev/null &
root@adpetlin:~# jobs
[1]+  Running                  yes > /dev/null &
root@adpetlin:~#
```

Рисунок 4.17: yes

Переводим процесс с перенаправлением вывода в фоновый режим. Проверяем состояния заданий, обращая внимание на процессы, выполняющиеся в фоновом режиме.

```
adpetlin@adpetlin:~$ nohup sleep 1000 &  
[1] 4916  
nohup: ignoring input and appending output to 'nohup.out'  
adpetlin@adpetlin:~$ jobs  
[1]+  Running                  nohup sleep 1000 &  
adpetlin@adpetlin:~$
```

Рисунок 4.18: nohup

Запускаем процесс таким образом, чтобы он продолжал работу после закрытия терминала.

```
4916 ?          00:00:00 sleep  
4938 ?          00:00:00 kworker/2:1-events  
5023 ?          00:00:00 kworker/5:0-mm_percpu_wq  
5094 ?          00:00:00 ptysis  
5101 ?          00:00:00 ptysis-agent  
5155 pts/1      00:00:00 bash  
5187 pts/1      00:00:00 ps  
adpetlin@adpetlin:~$
```

Рисунок 4.19: ps -A

Закрываем и заново открываем терминал, проверяя продолжение работы процесса. Изучаем информацию о запущенных процессах с помощью системного монитора.


```

adpetlin@adpetlin:~$ yes > /dev/null &
[1] 5213
adpetlin@adpetlin:~$ yes > /dev/null &
[2] 5219
adpetlin@adpetlin:~$ yes > /dev/null &
[3] 5221
adpetlin@adpetlin:~$ kill -9 5213
[1] Killed yes > /dev/null
adpetlin@adpetlin:~$ kill -9 %2
[2]- Killed yes > /dev/null
adpetlin@adpetlin:~$ jobs
[3]+ Running yes > /dev/null &
adpetlin@adpetlin:~$

```

Рисунок 4.20: yes

Запускаем три дополнительных фоновых процесса с перенаправлением вывода. Завершаем два процесса разными способами: по идентификатору процесса и по идентификатору задания.

```

adpetlin@adpetlin:~$ kill -1 4916
adpetlin@adpetlin:~$ kill -1 5221
[3]+ Hangup yes > /dev/null
adpetlin@adpetlin:~$

```

Рисунок 4.21: kill -1

Отправляем сигнал завершения процессу, запущенному с защитой от разрыва связи, и обычному процессу, сравнивая их поведение.

```

adpetlin@adpetlin:~$ jobs
[1]    Running                  yes > /dev/null &
[2]    Running                  yes > /dev/null &
[3]    Running                  yes > /dev/null &
[4]    Running                  yes > /dev/null &
[5]    Running                  yes > /dev/null &
[6]    Running                  yes > /dev/null &
[7]    Running                  yes > /dev/null &
[8]    Running                  yes > /dev/null &
[9]    Running                  yes > /dev/null &
[10]   Running                  yes > /dev/null &
[11]-  Running                  yes > /dev/null &
[12]+  Running                  yes > /dev/null &
adpetlin@adpetlin:~$ killall yes
[1]    Terminated             yes > /dev/null
[2]    Terminated             yes > /dev/null
[3]    Terminated             yes > /dev/null
[4]    Terminated             yes > /dev/null
[5]    Terminated             yes > /dev/null
[7]    Terminated             yes > /dev/null
[8]    Terminated             yes > /dev/null
[9]    Terminated             yes > /dev/null
[10]   Terminated             yes > /dev/null
[11]-  Terminated             yes > /dev/null
[12]+  Terminated             yes > /dev/null
[6]+   Terminated             yes > /dev/null
adpetlin@adpetlin:~$ █

```

Рисунок 4.22: killall

Запускаем несколько дополнительных фоновых процессов. Завершаем все процессы одновременно с помощью команды группового завершения.

```
5649  14   5 yes
5675  19   0 ps
adpetlin@adpetlin:~$
```

Рисунок 4.23: yes

Запускаем два процесса с разными приоритетами и сравниваем их абсолютные и относительные приоритеты.

```
5649   0  19 yes
5731  19   0 kworker/4:1-events
5747  19   0 kworker/u29:1-events_unbound
5770  19   0 yes
```

Рисунок 4.24: renice

Выравниваем приоритеты двух процессов, изменяя значение приоритета для одного из них.

5 Выводы

Мы получили навыки управления процессами операционной системы.

Список литературы

1. Поттеринг Л. Systemd для администраторов: цикл статей. — 2010. — URL: <http://wiki.opennet.ru/Systemd>.
2. Neil N. J. Learning CentOS: A Beginners Guide to Learning Linux. — CreateSpace Independent Publishing Platform, 2016.
3. Systemd. — 2022. — URL: <https://wiki.archlinux.org/title/Systemd>.