Отчёт по лабораторной работе №6

Управление процессами

Максат Хемраев

Содержание

Список иллюстраций

Список таблиц

1 Цель работы

Получить навыки управления процессами операционной системы.

2 Отчёт по выполнению работы

2.1 Управление заданиями в Linux

1. Вошёл в систему с полномочиями администратора с помощью команды **su**.
2. Запустил три задания:
   - **sleep 3600** в фоновом режиме;
   - **dd if=/dev/zero of=/dev/null** в фоновом режиме;
   - **sleep 7200** — без указания &, поэтому оболочка ожидала завершения процесса.

   Для приостановки последней команды применил комбинацию **Ctrl+Z**.

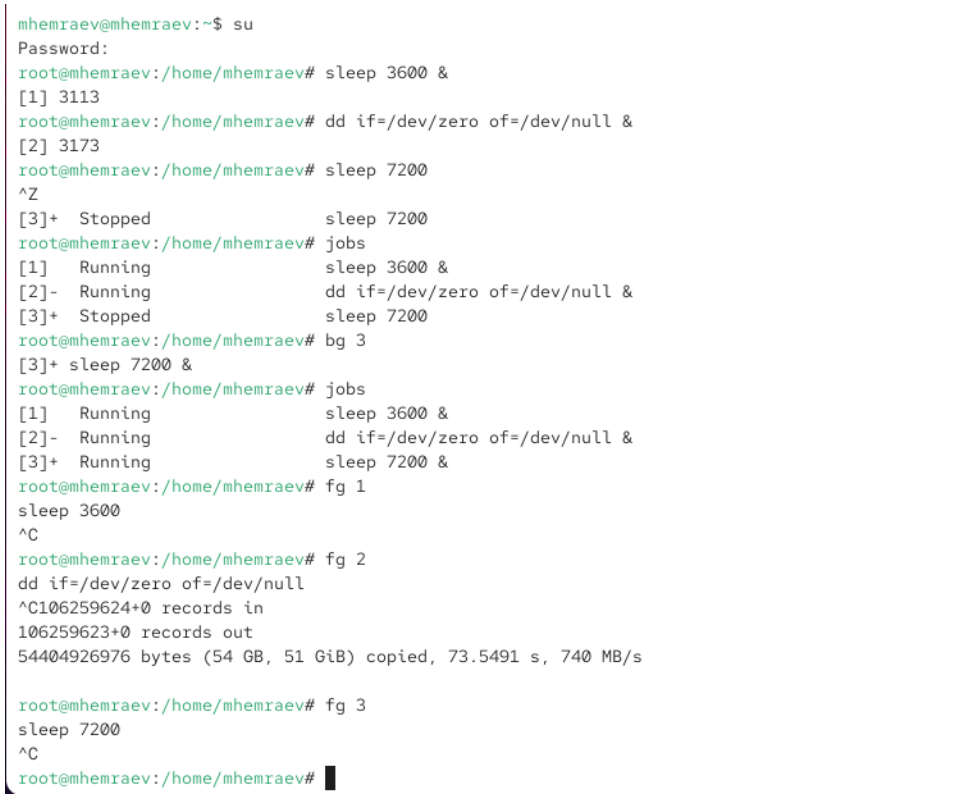

Рис. 2.1: Запуск и приостановка процессов

3. С помощью команды **jobs** проверил список заданий. Первые два процесса находились в состоянии *Running*, а третий — в состоянии *Stopped*.

4. Перевёл задание 3 в фоновый режим с помощью команды **bg 3**.
   После повторной проверки с помощью **jobs** убедился, что все три задания работают в фоне.

5. Перенёс задание 1 на передний план командой **fg 1**, затем завершил его с помощью **Ctrl+C**. Аналогично были завершены задания 2 и 3.

6. В новом терминале под обычным пользователем запустил команду **dd if=/dev/zero of=/dev/null &**.
   После этого закрыл терминал командой **exit**.

7. В другом терминале под своей учётной записью запустил мониторинг процессов с помощью утилиты **top**. В списке был виден процесс **dd**, который

продолжал выполняться.

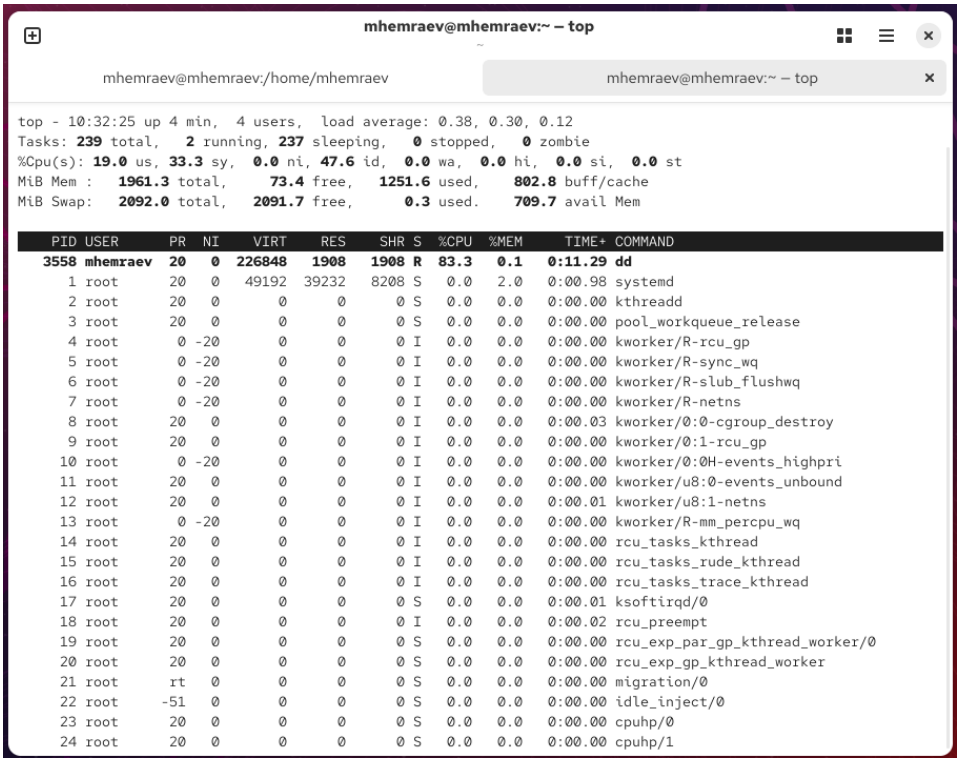

Рис. 2.2: Процесс dd в списке top

8. Повторно запустил **top** и завершил задание **dd** с помощью горячей клавиши **k**, указав PID процесса. После завершения процесса покинул утилиту командой **q**.

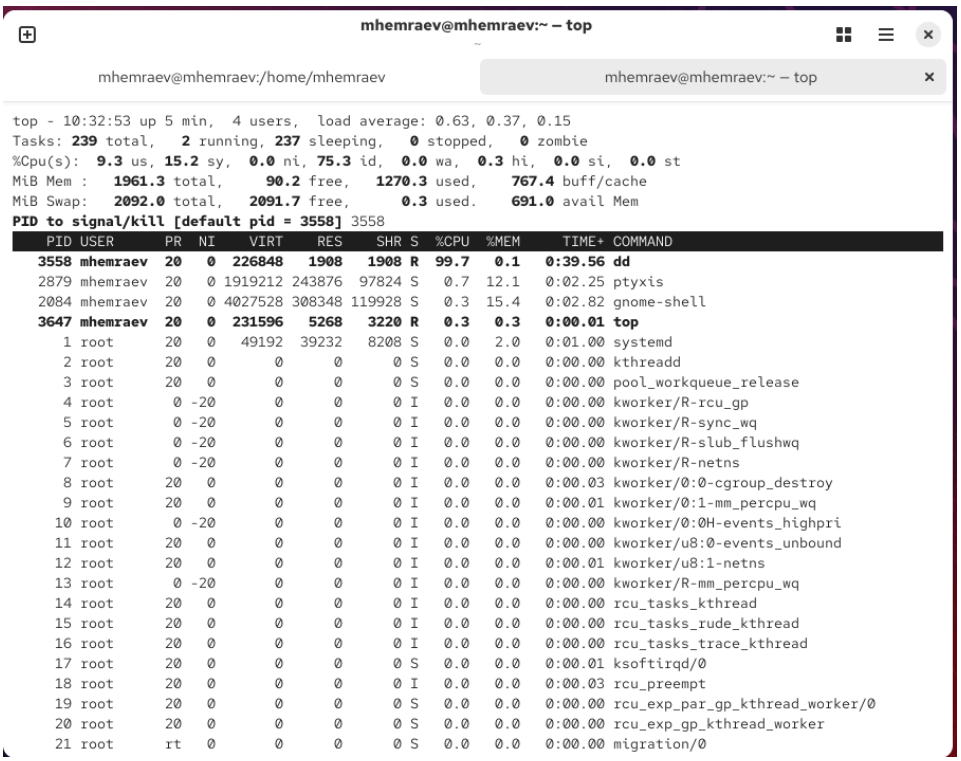

Рис. 2.3: Завершение процесса dd в top

2.2 Управление процессами в Linux

1. Получил полномочия администратора с помощью команды **su**.

2. Запустил три ресурсоёмких процесса в фоне:

   - **dd if=/dev/zero of=/dev/null &** (трижды).

3. Проверил список активных процессов, отфильтровав только те, что содержат **dd**, с помощью команды **ps aux | grep dd**. В выводе отобразились PID всех запущенных процессов **dd**.

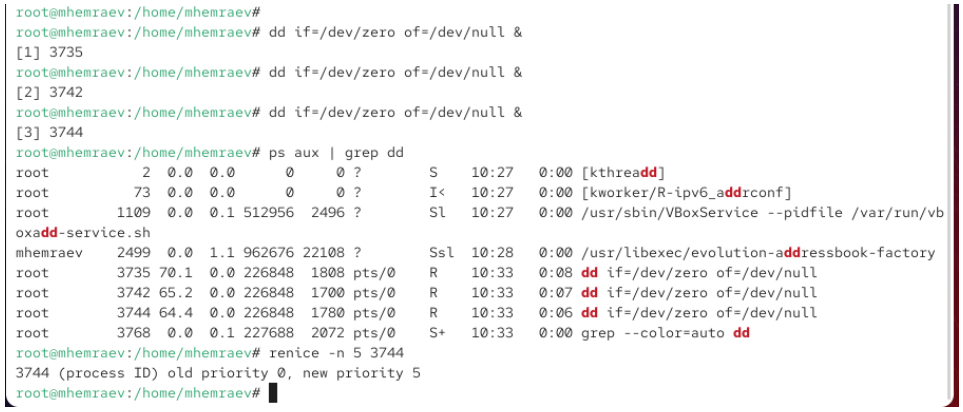

Рис. 2.4: Список процессов dd

4. Изменил приоритет одного из процессов с помощью команды **renice -n 5 <PID>**, что позволило задать новый приоритет.

5. Выполнил команду **ps fax | grep -B5 dd**, чтобы просмотреть дерево процессов. В выводе отобразилась иерархия, где процессы **dd** были запущены из родительской оболочки **bash**, запущенной от имени суперпользователя.

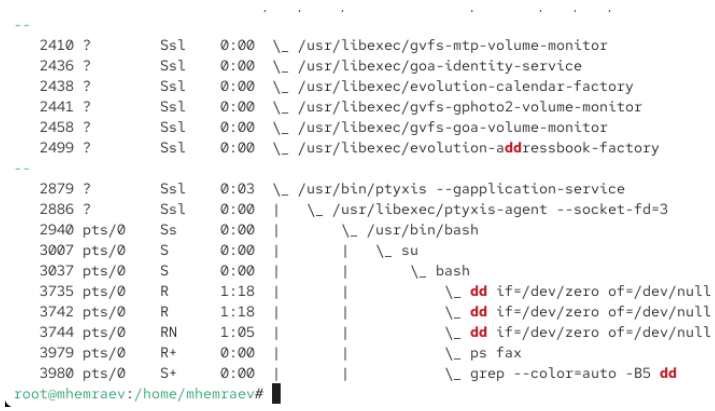

Рис. 2.5: Иерархия процессов dd

6. Определил PID корневой оболочки и завершил её вместе с дочерними процессами с помощью команды **kill -9 <PID>**. В результате оболочка закрылась, а все процессы **dd** были автоматически завершены.

2.3 Задание 1. Управление приоритетами и завершением процессов

1. Получил полномочия администратора с помощью команды **su**.
2. Запустил три фоновых процесса командой **dd if=/dev/zero of=/dev/null &**. Каждый процесс получил свой PID.
3. Изменил приоритет одного из процессов:
   - сначала задал значение **-5** с помощью команды **renice -n 5 <PID>**;
   - затем изменил тот же процесс на приоритет **-15**.

   Разница заключается в том, что чем меньше (и более отрицательное) значение nice, тем выше приоритет выполнения процесса. При **-5** процесс получает больше ресурсов процессора по сравнению с обычными задачами, а при **-15** его приоритет становится ещё выше, что может повлиять на производительность остальных процессов системы.
4. Завершил все запущенные процессы **dd**, используя команду **kill <PID>** для каждого из них.

```
mhemraev@mhemraev:~$ su
Password:
root@mhemraev:/home/mhemraev# dd if=/dev/zero of=/dev/null &
[1] 4396
root@mhemraev:/home/mhemraev# dd if=/dev/zero of=/dev/null &
[2] 4408
root@mhemraev:/home/mhemraev# dd if=/dev/zero of=/dev/null &
[3] 4410
root@mhemraev:/home/mhemraev# renice -n 5 4408
4408 (process ID) old priority 0, new priority 5
root@mhemraev:/home/mhemraev# renice -n 15 4408
4408 (process ID) old priority 5, new priority 15
root@mhemraev:/home/mhemraev# kill 4396
root@mhemraev:/home/mhemraev#
[1]   Terminated              dd if=/dev/zero of=/dev/null
root@mhemraev:/home/mhemraev# kill 4408
[2]-  Terminated              dd if=/dev/zero of=/dev/null
root@mhemraev:/home/mhemraev#
root@mhemraev:/home/mhemraev# kill 4410
[3]+  Terminated              dd if=/dev/zero of=/dev/null
root@mhemraev:/home/mhemraev#
root@mhemraev:/home/mhemraev#
```

Рис. 2.6: Управление приоритетами и завершение процессов

2.4 Задание 2. Управление процессами с использованием программы *yes*

1. Запустил программу **yes** в фоновом режиме с перенаправлением вывода в **/dev/null**.

2. Запустил **yes** на переднем плане с подавлением вывода. Остановил выполнение комбинацией **Ctrl+Z**, затем вновь запустил и завершил процесс.

```
root@mhemraev:/home/mhemraev#
root@mhemraev:/home/mhemraev# yes > /dev/null &
[1] 4699
root@mhemraev:/home/mhemraev# yes > /dev/null
^Z
[2]+  Stopped                 yes > /dev/null
root@mhemraev:/home/mhemraev# yes > /dev/null
^C
root@mhemraev:/home/mhemraev#
```

Рис. 2.7: Запуск и остановка yes

3. Запустил **yes** на переднем плане без подавления вывода. Приостановил

выполнение, затем продолжил и завершил процесс.

4. Проверил состояния процессов с помощью команды **jobs** — отобразились все текущие задания и их статусы (*Running* или *Stopped*).

```
root@mhemraev:/home/mhemraev# jobs
[1]-  Running                 yes > /dev/null &
[2]+  Stopped                 yes > /dev/null
root@mhemraev:/home/mhemraev# fg 1
yes > /dev/null
^C
root@mhemraev:/home/mhemraev# jobs
[2]+  Stopped                 yes > /dev/null
root@mhemraev:/home/mhemraev# bg 2
[2]+ yes > /dev/null &
root@mhemraev:/home/mhemraev#
root@mhemraev:/home/mhemraev# jobs
[2]+  Running                 yes > /dev/null &
root@mhemraev:/home/mhemraev# nohup yes > /dev/null &
[3] 4944
nohup: ignoring input and redirecting stderr to stdout
root@mhemraev:/home/mhemraev#
root@mhemraev:/home/mhemraev# jobs
[2]-  Running                 yes > /dev/null &
[3]+  Running                 nohup yes > /dev/null &
root@mhemraev:/home/mhemraev#
```

Рис. 2.8: Состояния заданий

5. Перевёл один из процессов на передний план (**fg**), после чего остановил его.

6. Перевёл другой процесс с подавлением вывода в фоновый режим командой **bg**, что изменило его статус на *Running*.

7. Запустил процесс в фоне с помощью **nohup yes > /dev/null &**, что позволило сохранить его выполнение даже после выхода из терминала. Проверка показала, что процесс продолжил работу.

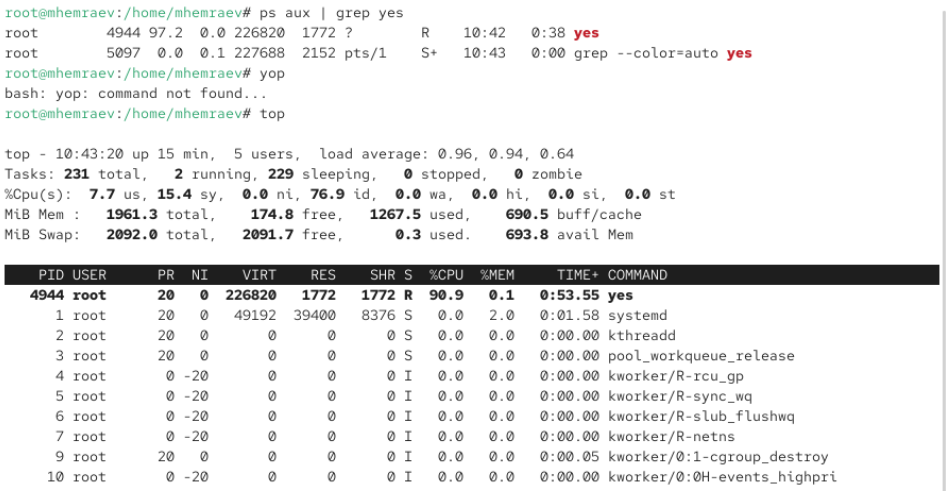

Рис. 2.9: Работа nohup-процесса

8. Получил информацию о процессах с помощью утилиты **top**, где был виден работающий процесс **yes**.

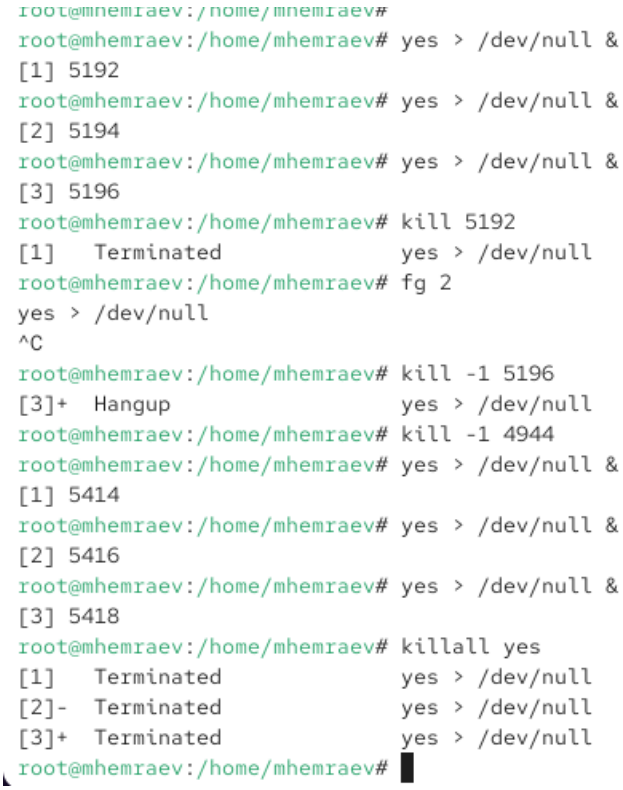

Рис. 2.10: Процесс yes в top

9. Запустил ещё три процесса **yes** в фоне с подавлением вывода. Завершил

их разными способами: один через **kill <PID>**, другой через **kill -<номер задания>**, а оставшиеся командой **killall yes**.

```
root@mhemraev:/home/mhemraev# yes > /dev/null &
[1] 5678
root@mhemraev:/home/mhemraev# nice -n 5 yes > /dev/null &
[2] 5691
root@mhemraev:/home/mhemraev# ps -l
F S   UID     PID    PPID  C PRI  NI ADDR SZ WCHAN  TTY          TIME CMD
4 S     0    5016    4993  0  80   0 - 58153 do_wai pts/1    00:00:00 su
4 S     0    5028    5016  0  80   0 - 57543 do_wai pts/1    00:00:00 bash
4 R     0    5678    5028 98  80   0 - 56705 -      pts/1    00:00:15 yes
4 R     0    5691    5028 97  85   5 - 56705 -      pts/1    00:00:07 yes
4 R     0    5713    5028  0  80   0 - 57682 -      pts/1    00:00:00 ps
root@mhemraev:/home/mhemraev# renice -n 5 5678
5678 (process ID) old priority 0, new priority 5
root@mhemraev:/home/mhemraev# ps -l
F S   UID     PID    PPID  C PRI  NI ADDR SZ WCHAN  TTY          TIME CMD
4 S     0    5016    4993  0  80   0 - 58153 do_wai pts/1    00:00:00 su
4 S     0    5028    5016  0  80   0 - 57543 do_wai pts/1    00:00:00 bash
4 R     0    5678    5028 98  85   5 - 56705 -      pts/1    00:00:40 yes
4 R     0    5691    5028 97  85   5 - 56705 -      pts/1    00:00:32 yes
4 R     0    5771    5028  0  80   0 - 57682 -      pts/1    00:00:00 ps
root@mhemraev:/home/mhemraev#
```

Рис. 2.11: Завершение процессов yes

10. Запустил два процесса **yes**: один с обычным приоритетом, второй с повышенным приоритетом с помощью **nice -n 5**. Сравнил приоритеты с помощью **ps -l**: значения поля **NI** (nice value) оказались разными.

11. Изменил приоритет одного из процессов с помощью команды **renice**, чтобы оба процесса имели одинаковое значение приоритета.

3 Контрольные вопросы

3.0.1 1. Какая команда даёт обзор всех текущих заданий оболочки?

- Команда **jobs** отображает список всех запущенных заданий в текущей оболочке с указанием их состояния (*Running*, *Stopped* и т.д.).

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

- Для остановки используется комбинация **Ctrl+Z**.

- Чтобы возобновить выполнение задания в фоне — команда **bg <номер_задания>**.

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

- Комбинация **Ctrl+C** завершает текущее активное задание.

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

- Использовать команду **kill <PID>** или **killall <имя_процесса>** из другой оболочки.

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

- Команда **ps fax** (или **pstree**) показывает дерево процессов с иерархией.

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

- Используется команда **renice** с отрицательным значением nice (чем меньше число, тем выше приоритет).

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

- Использовать команду **killall dd**, которая завершает все процессы с именем **dd**.

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

- Для этого используется команда **killall mycommand**.

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

- В **top** для завершения процесса используется клавиша **k**, после чего указывается PID процесса.

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

- Для запуска с пониженным приоритетом (что гарантирует ресурс другим процессам) используется команда **nice** с положительным значением.

---

4 Заключение

В ходе работы были изучены основные приёмы управления процессами и заданиями в Linux. Рассмотрены способы запуска процессов в фоновом и переднем планах, их приостановка, возобновление и завершение. Освоены инструменты изменения приоритета процессов с помощью утилит **nice** и **renice**, а также методы массового завершения процессов через **killall**. Дополнительно были изучены средства мониторинга системных ресурсов с помощью **top** и команды для анализа иерархии процессов. Полученные знания позволяют эффективно контролировать работу процессов и управлять их ресурсами в многозадачной среде.