Отчёт по лабораторной работе №6

Управление процессами

Яковлева Дарья Сергеевна

Содержание

Список иллюстраций

Список таблиц

1 Цель работы

Получить навыки управления процессами операционной системы.

2 Выполнение лабораторной работы

2.1 Управление заданиями

Получаю полномочия администратора с помощью `su` (см. рис. fig. 2.1).

Запускаю три задания:

- `sleep 3600 &` — первое задание в фоне;
- `dd if=/dev/zero of=/dev/null &` — второе задание в фоне;
- `sleep 7200` — третье задание, запущенное без `&`.

Поскольку третье задание блокирует терминал, прерываю его комбинацией **Ctrl+Z**, после чего проверяю список заданий с помощью `jobs` (см. рис. fig. 2.1).

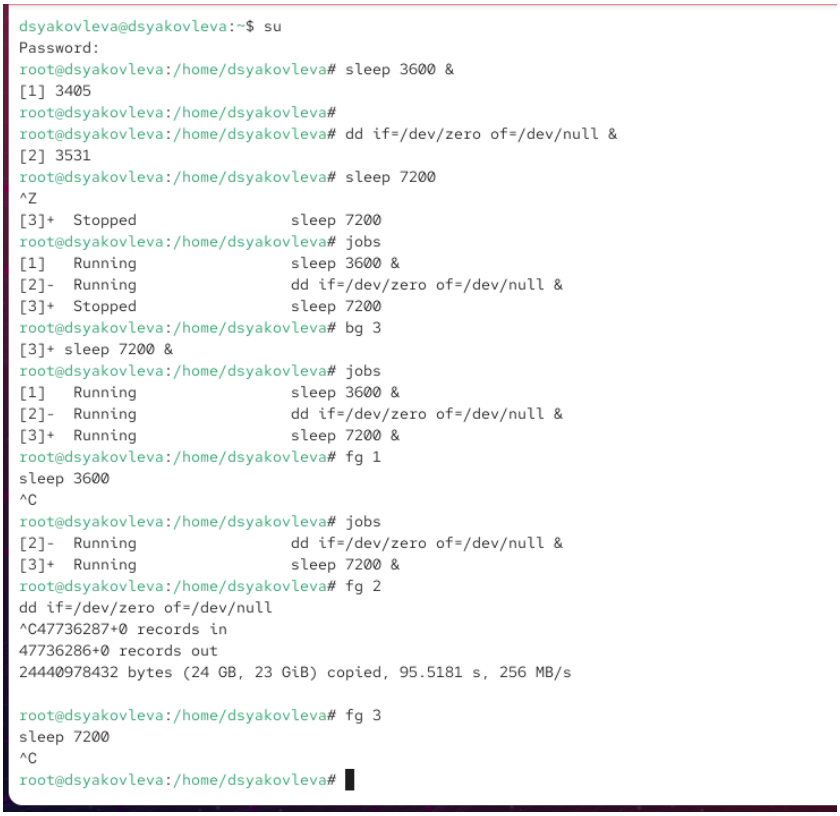

Рис. 2.1: Запуск и управление заданиями

Вижу, что первые два задания находятся в состоянии *Running*, а третье — *Stopped*. Перевожу задание 3 в фоновый режим командой `bg 3`, снова проверяю список заданий через `jobs`. Теперь все три задания работают в фоне.

Переношу задание 1 на передний план с помощью `fg 1`, затем останавливаю его комбинацией **Ctrl+C**. Аналогично завершаю задания 2 и 3 (см. рис. fig. 2.1).

Открываю новый терминал от имени пользователя и запускаю в нём процесс `dd if=/dev/zero of=/dev/null &`.
Закрываю терминал командой `exit`.

В другом терминале под тем же пользователем запускаю утилиту `top`, где вижу, что процесс `dd` продолжает работать (см. рис. fig. 2.2).

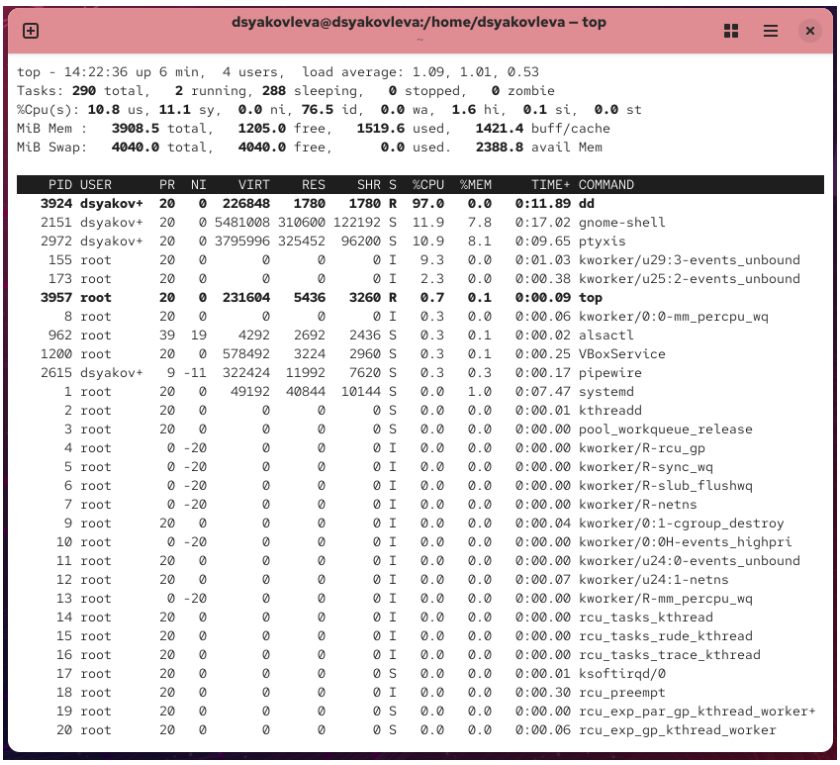

Рис. 2.2: Работа с утилитой top

Выход из `top` выполняю клавишей **q**. Затем снова запускаю `top` и использую клавишу **k** для завершения процесса `dd`. В списке процессов он больше не отображается (см. рис. fig. 2.3).

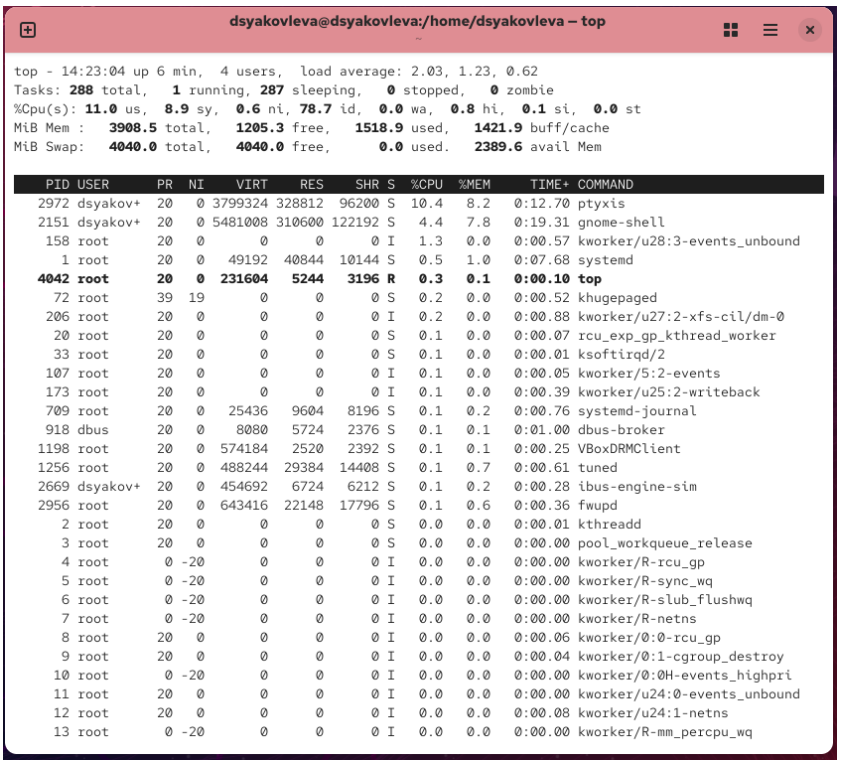

Рис. 2.3: Завершение процесса dd

2.2 Управление процессами

Получаю полномочия администратора с помощью su.

Запускаю три процесса dd, которые пишут поток данных из /dev/zero в /dev/null (см. рис. fig. 2.4).

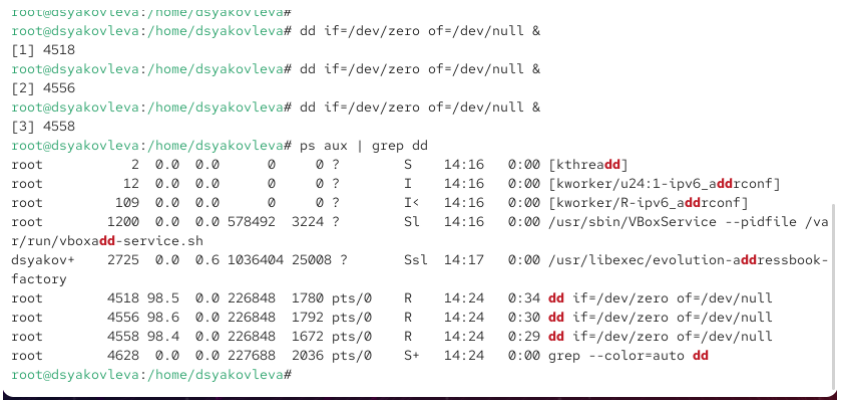

Рис. 2.4: Запуск процессов dd

С помощью команды `ps aux | grep dd` просматриваю список процессов. Вижу три активных процесса dd с различными PID (см. рис. fig. 2.4).

Меняю приоритет одного из процессов, используя команду `renice -n 5 <PID>`. В результате приоритет изменяется с 0 на 5 (см. рис. fig. 2.5).

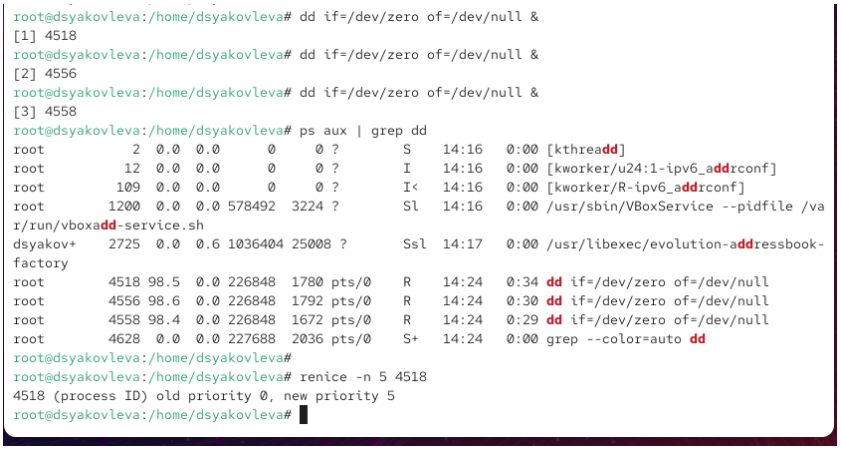

Рис. 2.5: Изменение приоритета процесса

Для отображения иерархии процессов использую команду `ps fax | grep -B5 dd`. Вижу дерево выполнения, где процессы dd являются дочерними для корневой оболочки bash, запущенной из-под su (см. рис. fig. 2.6).

Рис. 2.6: Просмотр иерархии процессов

Нахожу PID родительского процесса bash, из которого были запущены процес-

сы dd, и завершаю его с помощью `kill -9 <PID>`. В результате оболочка закрывается, и все дочерние процессы dd останавливаются автоматически.

2.3 Задание 1

Запускаю три процесса dd, которые пишут данные из `/dev/zero` в `/dev/null`, в фоновом режиме (см. рис. fig. 2.7).

```
root@dsyakovleva:/home/dsyakovleva#
root@dsyakovleva:/home/dsyakovleva# dd if=/dev/zero of=/dev/null &
[1] 5279
root@dsyakovleva:/home/dsyakovleva# dd if=/dev/zero of=/dev/null &
[2] 5288
root@dsyakovleva:/home/dsyakovleva# dd if=/dev/zero of=/dev/null &
[3] 5293
root@dsyakovleva:/home/dsyakovleva# renice -n 5 5288
5288 (process ID) old priority 0, new priority 5
root@dsyakovleva:/home/dsyakovleva# renice -n 15 5288
5288 (process ID) old priority 5, new priority 15
root@dsyakovleva:/home/dsyakovleva# 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@dsyakovleva:/home/dsyakovleva#
```

Рис. 2.7: Запуск процессов dd

Меняю приоритет одного из процессов на значение `-5` с помощью команды `renice -n -5 <PID>`. В результате приоритет изменяется с 0 на 5.

Затем повторно изменяю приоритет этого же процесса, установив значение `-15`. Теперь приоритет изменяется с 5 на 15 (см. рис. fig. 2.7).

Для завершения всех процессов dd использую команду `killall dd`. Вижу сообщение о завершении трёх процессов (см. рис. fig. 2.7).

2.4 Задание 2

Запускаю программу `yes` в фоновом режиме с подавлением потока вывода. Аналогично запускаю её несколько раз на переднем плане, приостанавливаю выполнение комбинацией **Ctrl+Z**, затем перевожу процессы в фон с помощью `bg` и завершаю их при необходимости (см. рис. fig. 2.8).

Рис. 2.8: Запуск программы yes

С помощью команды `jobs` проверяю состояния заданий. Вижу, что процессы находятся в состоянии *Running*. Один из процессов перевожу на передний план командой `fg`, затем завершаю его (см. рис. fig. 2.9).

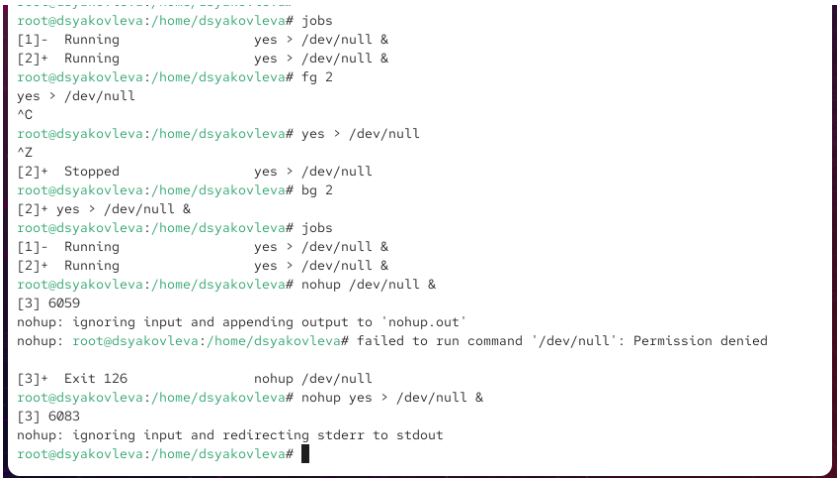

Рис. 2.9: Управление заданиями yes

Для запуска процесса, продолжающего работу после выхода из терминала, использую `nohup yes > /dev/null &`. Проверяю результат через `jobs`, где видно, что процесс перенаправил вывод в файл `nohup.out`. После закрытия терминала и повторного входа убеждаюсь, что процесс остался активным (см. рис. fig. 2.9).

Получаю информацию о процессах с помощью утилиты `top`. Наглядно видно, что процессы `yes` потребляют значительные ресурсы процессора (см. рис. fig. 2.10).

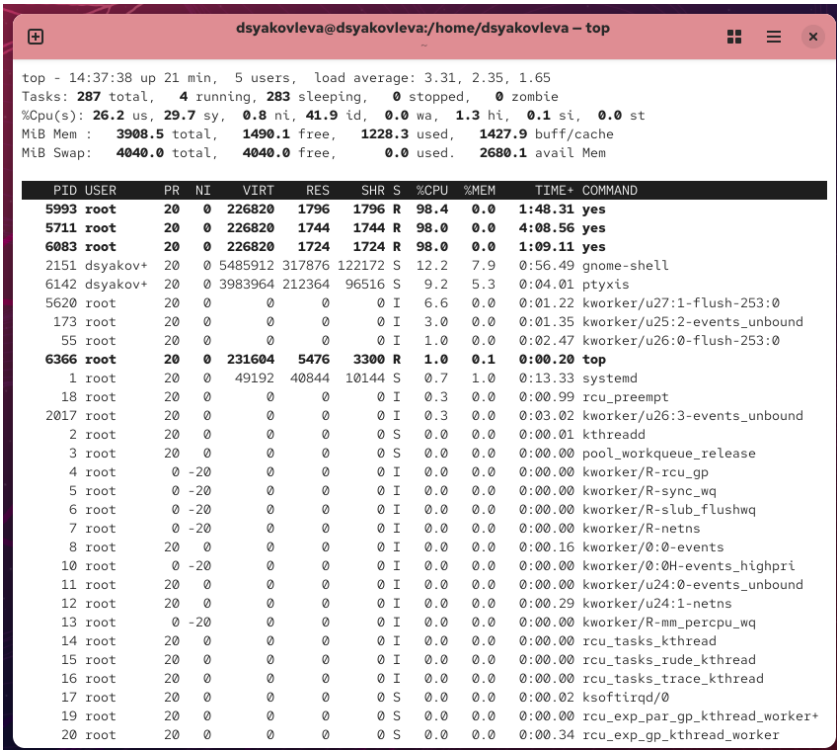

Рис. 2.10: Процессы yes в top

Запускаю ещё три процесса `yes` в фоне. Завершаю один из них по PID, а второй — по идентификатору задания. Использую команду `kill` и наблюдаю завершение процессов (см. рис. fig. 2.11).

Рис. 2.11: Завершение процессов yes

Далее отправляю сигнал `SIGHUP` обычному процессу и процессу, запущенному с помощью `nohup`. При этом обычный процесс завершается, а процесс под `nohup` продолжает работу (см. рис. fig. 2.12).

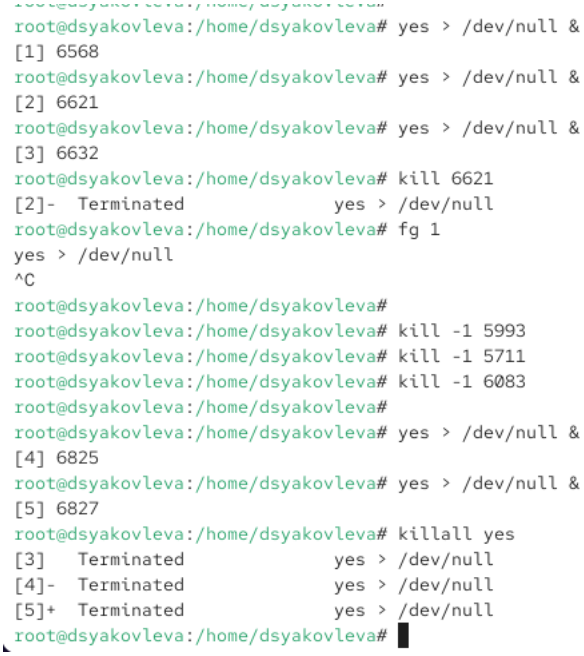

Рис. 2.12: Проверка работы с сигналами

Запускаю несколько процессов `yes` в фоне, а затем завершаю их одновременно командой `killall yes` (см. рис. fig. 2.13).

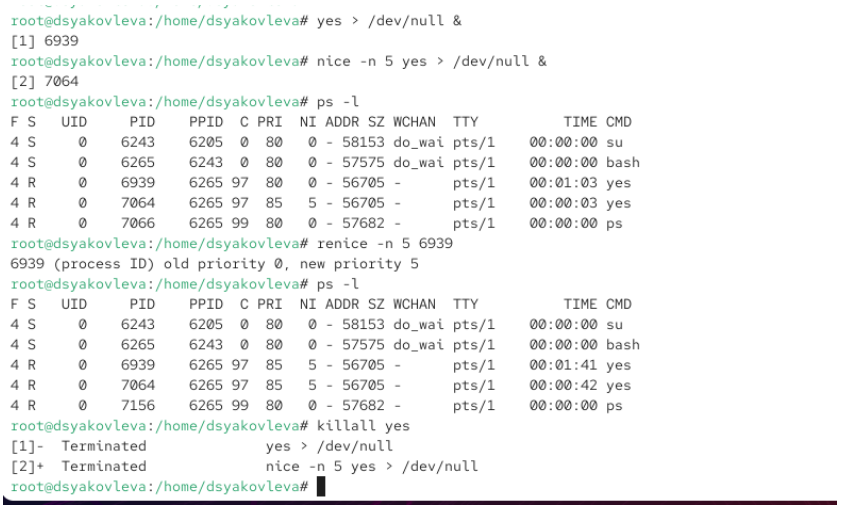

Рис. 2.13: Завершение процессов командой killall

Запускаю процесс `yes` с повышенным приоритетом при помощи `nice -n -5`. Сравниваю приоритеты процессов через `ps -l`: вижу различие в значениях. Затем с помощью `renice` выравниваю приоритеты, установив одинаковое значение

для обоих процессов (см. рис. fig. 2.13).

3 Контрольные вопросы

1. **Какая команда даёт обзор всех текущих заданий оболочки?**
   Используется команда `jobs`.

2. **Как остановить текущее задание оболочки, чтобы продолжить его выполнение в фоновом режиме?**
   Используется комбинация клавиш **Ctrl+Z**, затем команда bg.

3. **Какую комбинацию клавиш можно использовать для отмены текущего задания оболочки?**
   Для завершения задания применяется комбинация **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 <PID>` из другой оболочки или завершить родительский процесс, от которого оно запущено.

5. **Какая команда используется для отображения отношений между родительскими и дочерними процессами?**
   Команда `ps fax`.

6. **Какая команда позволит изменить приоритет процесса с идентификатором 1234 на более высокий?**
   Команда `renice -n -5 -p 1234`.

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

   Использовать команду `killall dd`.

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

   Используется команда `killall mycommand`.

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

   Внутри `top` используется клавиша **k** и вводится PID процесса.

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

    Используется команда `nice`, например: `nice -n 10 mycommand`.

4 Заключение

В ходе лабораторной работы были приобретены практические навыки управления заданиями и процессами в Linux: запуск программ в фоновом и переднем режиме, приостановка и возобновление выполнения, завершение отдельных процессов и групп процессов. Освоены приёмы работы с приоритетами процессов с помощью утилит `nice` и `renice`, а также использование команд `jobs`, `ps`, `top`, `kill` и `killall`. Полученные знания позволяют эффективно контролировать работу процессов и управлять распределением ресурсов в системе.