Отчёт по лабораторной работе №6

Управление процессами

Эзиз Хатамов

Содержание

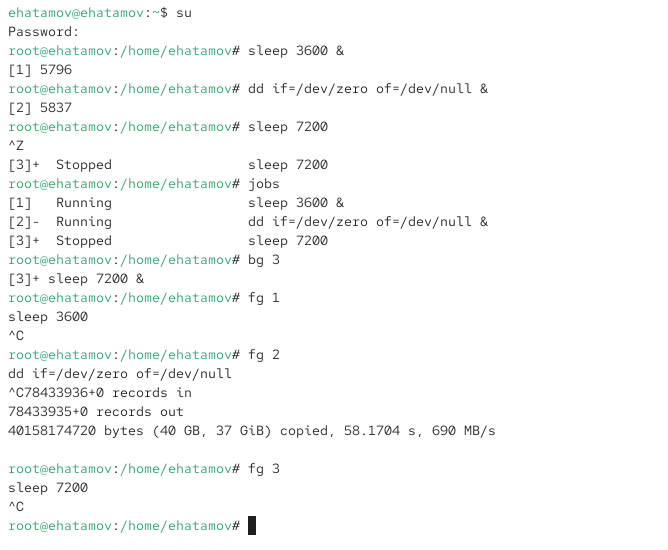
# 1 Цель работы

Получить навыки управления процессами операционной системы.

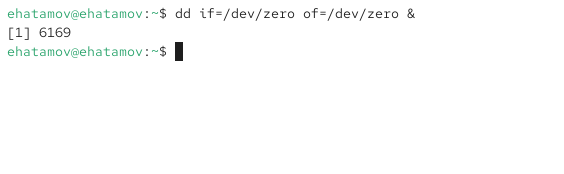
# 2 Отчёт по выполнению работы

## 2.1 Управление заданиями

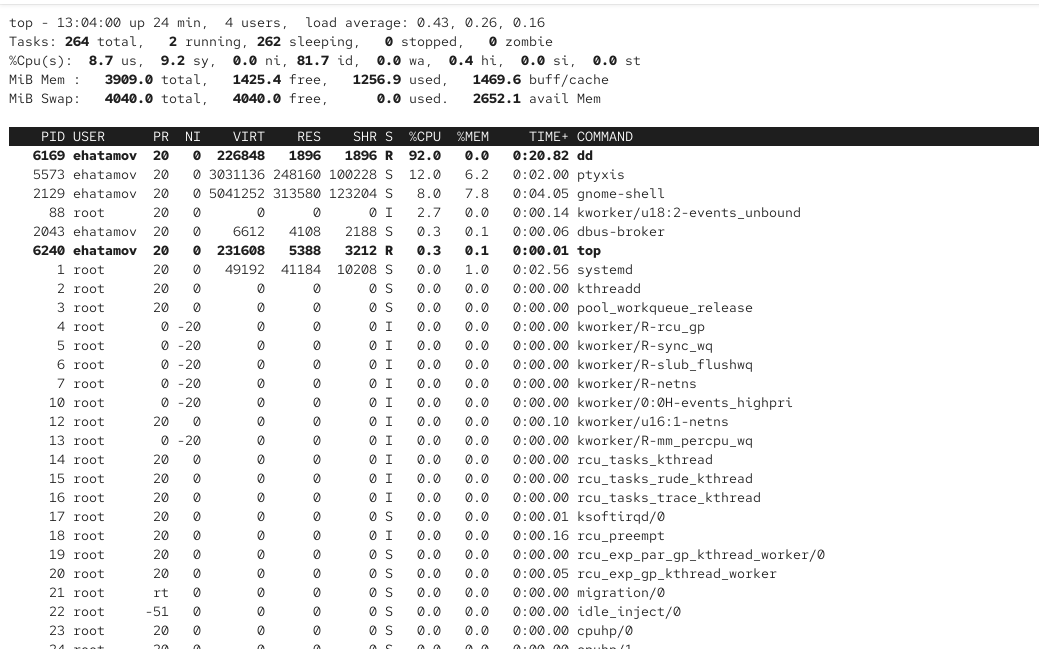
1. Пользователь получил полномочия администратора с помощью команды su -.
2. В фоновом режиме были запущены команды:
   * sleep 3600 &
   * dd if=/dev/zero of=/dev/null &  
     Дополнительно была запущена команда sleep 7200 без амперсанда.
3. Так как команда sleep 7200 выполнялась на переднем плане, для остановки её работы использовалось сочетание клавиш **Ctrl+Z**.
4. Команда jobs показала три активных задания:
   * задание 1 и 2 находились в состоянии **Running**,
   * задание 3 — в состоянии **Stopped**.
5. Для возобновления выполнения задания 3 в фоновом режиме была выполнена команда bg 3.  
   Статус изменился на **Running**.
6. Перемещение задания 1 на передний план осуществлялось командой fg 1.  
   Далее оно было прервано комбинацией **Ctrl+C**.

* 
* Рис. 1: Управление заданиями

1. Аналогично задания 2 и 3 были выведены на передний план (fg 2, fg 3) и прерваны комбинацией **Ctrl+C**.
2. В другом терминале под обычным пользователем было запущено dd if=/dev/zero of=/dev/null &.  
   Процесс успешно стартовал в фоне.

* 
* Рис. 2: Фоновый процесс

1. После выхода из второго терминала команда top в другом окне показала, что процесс **dd** продолжает выполняться.

* 
* Рис. 3: Отображение фонового процесса dd в top

1. Для завершения процесса в top использовалась клавиша **k**, после чего задание **dd** было остановлено.

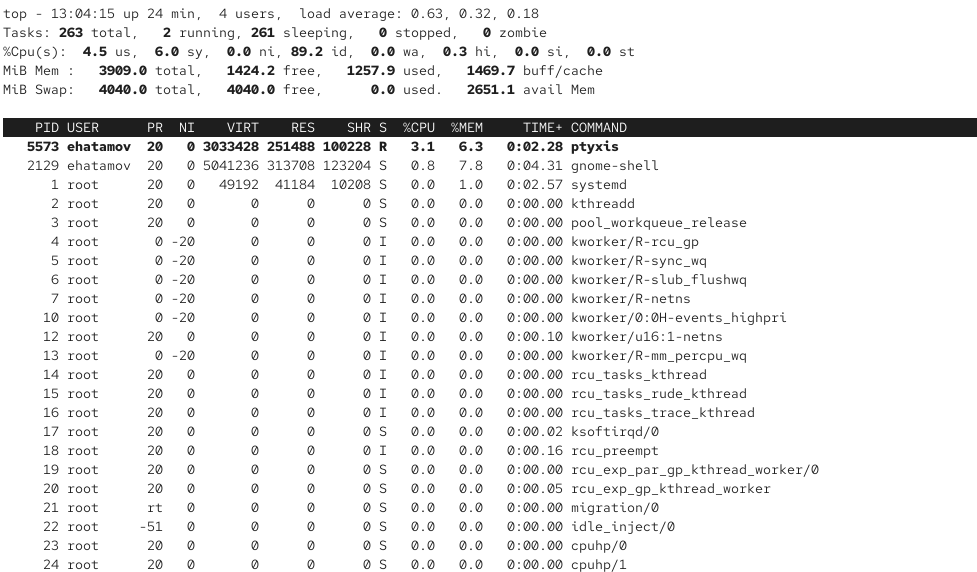
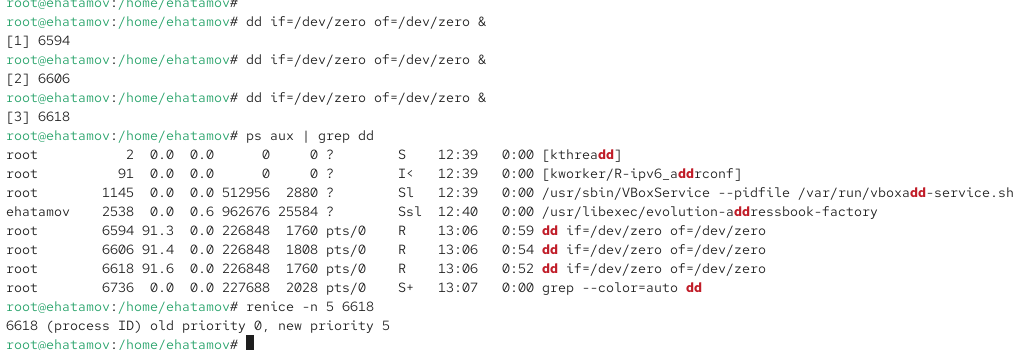


Рис. 4: Завершение работы процесса dd в top

## 2.2 Управление процессами

1. Пользователь получил полномочия администратора с помощью команды su -.
2. В фоновом режиме были запущены три процесса:
   * dd if=/dev/zero of=/dev/null &
   * dd if=/dev/zero of=/dev/null &
   * dd if=/dev/zero of=/dev/null &
3. Для проверки списка запущенных процессов использовалась команда ps aux | grep dd.  
   В результате были отображены три активных процесса **dd**.

* 
* Рис. 5: Список процессов dd

1. Приоритет одного из процессов был изменён с помощью команды renice -n 5 <PID>.  
   В отчёте видно, что приоритет процесса с PID 6618 изменился с 0 на 5.
2. Для отображения иерархии процессов была использована команда ps fax | grep -B5 dd.  
   Она показала дерево процессов, включая родительскую оболочку, из которой были запущены все процессы **dd**.
3. Используя найденный PID родительской оболочки, была выполнена команда kill -9 <PID>.  
   В результате завершилась корневая оболочка и автоматически остановились все дочерние процессы **dd**.

* 
* Рис. 6: Завершение родительской оболочки и дочерних процессов dd

## 2.3 Задание 1. Управление приоритетами процессов

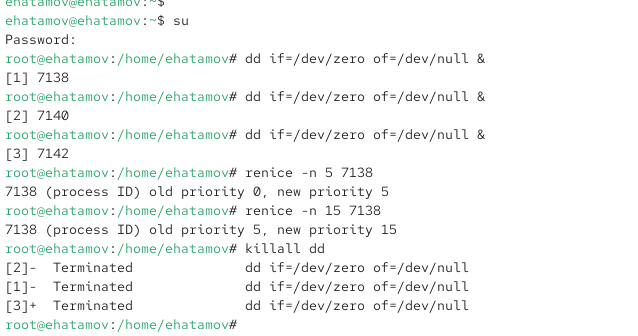
1. Пользователь получил права администратора с помощью команды su.
2. В фоновом режиме были запущены три процесса:
   * dd if=/dev/zero of=/dev/null &
   * dd if=/dev/zero of=/dev/null &
   * dd if=/dev/zero of=/dev/null &

* Каждый процесс получил собственный PID.

1. Приоритет одного из процессов был изменён командой renice -n 5 <PID>.  
   В результате его приоритет изменился с **0** на **5**.
2. Далее приоритет того же процесса был снова изменён:
   * команда renice -n 15 <PID> повысила значение приоритета до **15**.

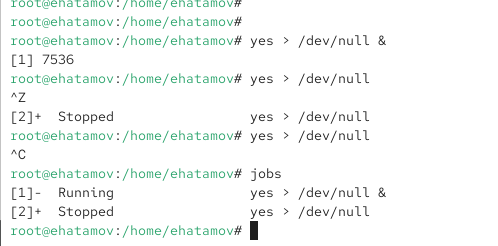
* Разница заключается в том, что:
  + **меньшее значение nice (например, −5)** повышает приоритет процесса, предоставляя ему больше процессорного времени;
  + **большее значение nice (например, 15)** понижает приоритет, снижая доступное процессорное время.

1. Для завершения всех запущенных процессов **dd** использовалась команда killall dd.  
   Все три процесса были успешно остановлены.

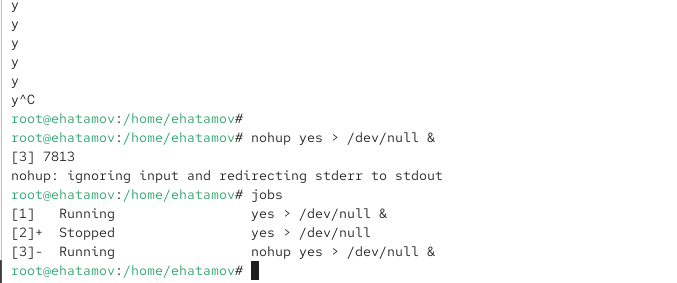
* 
* Рис. 7: Управление приоритетами и завершение процессов dd

## 2.4 Задание 2. Работа с процессами и заданиями

1. В фоновом режиме была запущена программа yes с подавлением потока вывода:
   * yes > /dev/null &
2. Программа yes была запущена на переднем плане с перенаправлением вывода.  
   С помощью **Ctrl+Z** выполнение было приостановлено, затем процесс возобновлён и завершён.
3. Программа yes была запущена на переднем плане без подавления потока вывода.  
   После нескольких выводов символа **y** её выполнение было приостановлено (**Ctrl+Z**), возобновлено и завершено (**Ctrl+C**).

* 
* Рис. 8: Работа с процессом yes на переднем плане

1. Проверка состояния заданий с помощью команды jobs показала активные и приостановленные процессы.
2. Один из процессов, выполнявшийся в фоне, был переведён на передний план командой fg и остановлен.
3. Процесс с подавлением вывода был переведён обратно в фоновый режим командой bg.
4. Повторная команда jobs подтвердила, что процесс находится в состоянии **Running** в фоне.
5. Для запуска процесса, продолжающего работу после выхода из терминала, использовалась команда:
   * nohup yes > /dev/null &

* 
* Рис. 9: Запуск процесса yes с nohup

1. После закрытия окна терминала процесс, запущенный через nohup, продолжил работу.
2. Для просмотра состояния процессов использовалась утилита top, где были видны процессы **yes**, нагружающие процессор.

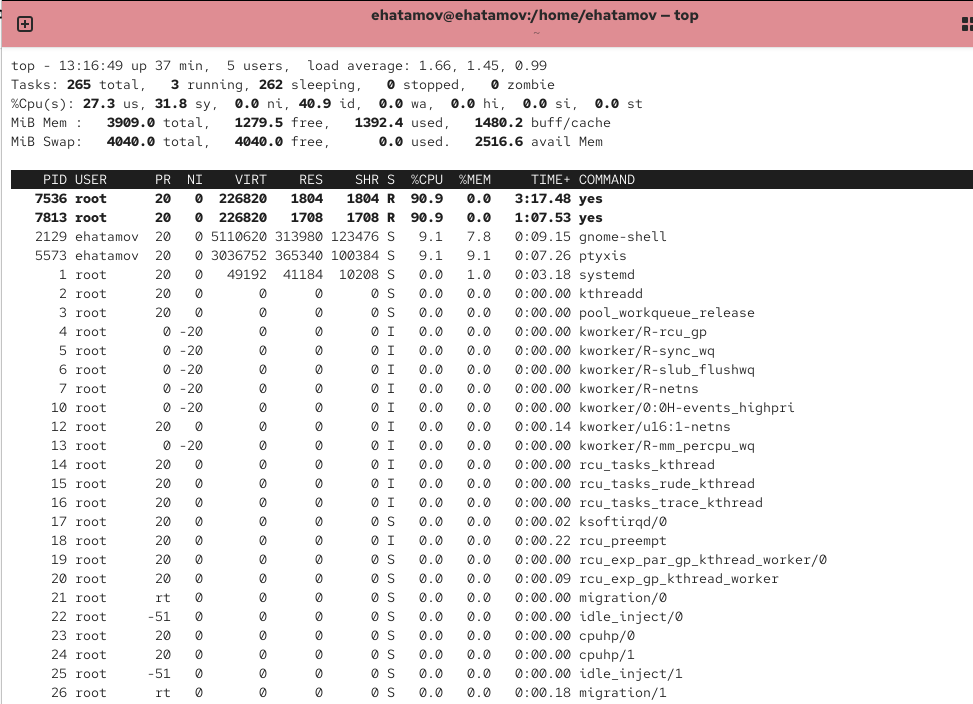


Рис. 10: Процессы yes в top

1. Дополнительно были запущены три программы yes в фоновом режиме.
2. Два процесса были завершены разными способами:

* по PID с помощью команды kill <PID>;
* по идентификатору задания с помощью команды kill %<номер\_задания>.

1. Была предпринята попытка отправить сигнал **SIGHUP (1)** процессу, запущенному с помощью nohup, и обычному процессу.

* процесс без nohup завершился;
* процесс с nohup продолжил работу.

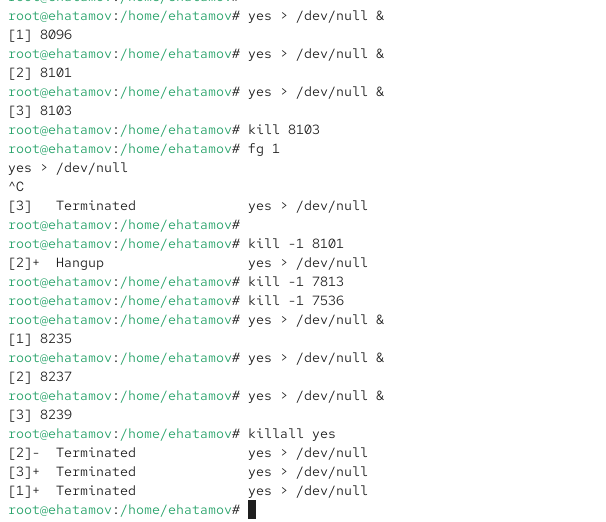


Рис. 11: Завершение процессов yes

1. Были запущены новые процессы yes в фоне.
2. Для их одновременного завершения использовалась команда:

* killall yes

1. Программа yes была запущена с разным приоритетом:

* yes > /dev/null & (обычный приоритет 0),
* nice -n 5 yes > /dev/null & (пониженный приоритет 5).

Сравнение в выводе ps -l показало различие в значениях **NI** (nice).

1. Приоритет одного из потоков был изменён командой renice, чтобы у обоих процессов он стал одинаковым.

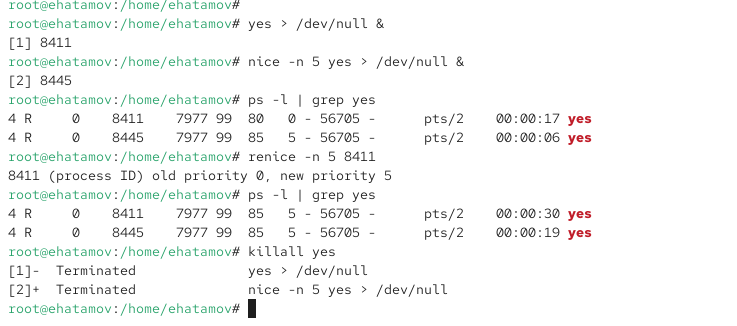


Рис. 12: Изменение приоритетов процессов yes

# 3 Контрольные вопросы

1. **Какая команда даёт обзор всех текущих заданий оболочки?**
   * jobs
   * Пример: jobs — выводит список всех фоновых и приостановленных заданий.
2. **Как остановить текущее задание оболочки, чтобы продолжить его выполнение в фоновом режиме?**
   * Ctrl+Z — приостановка выполнения.
   * bg — продолжение выполнения в фоне.
3. **Какую комбинацию клавиш можно использовать для отмены текущего задания оболочки?**
   * Ctrl+C — завершает текущее задание.
4. **Необходимо отменить одно из начатых заданий. Доступ к оболочке, в которой работает пользователь, невозможен. Что можно сделать?**
   * Использовать команду kill <PID> в другой оболочке.
   * Пример: kill 1234 — завершает процесс с PID **1234**.
5. **Какая команда используется для отображения отношений между родительскими и дочерними процессами?**
   * ps fax
   * Пример: ps fax | grep dd — покажет дерево процессов с фильтрацией по **dd**.
6. **Какая команда позволит изменить приоритет процесса с идентификатором 1234 на более высокий?**
   * renice -n -5 -p 1234 — повысить приоритет процесса.
7. **В системе в настоящее время запущено 20 процессов dd. Как проще всего остановить их все сразу?**
   * killall dd — завершает все процессы с именем **dd**.
8. **Какая команда позволяет остановить команду с именем mycommand?**
   * killall mycommand
9. **Какая команда используется в top, чтобы убить процесс?**
   * В top нажать клавишу k, затем указать **PID** процесса.
10. **Как запустить команду с достаточно высоким приоритетом, не рискуя, что не хватит ресурсов для других процессов?**
    * Использовать nice с положительным значением.
    * Пример: nice -n 10 ./script.sh — запускает команду с пониженным приоритетом, оставляя ресурсы другим процессам.

# 4 Заключение

В ходе работы были изучены команды для управления заданиями и процессами в Linux: jobs, bg, fg, kill, killall, ps fax, а также утилиты top, nice и renice. Эти инструменты позволяют контролировать выполнение процессов, управлять их приоритетами и корректно завершать работу в различных ситуациях.