Отчёт по лабораторной работе №6

Управление процессами

Шаханеоядж Хаоладар

Содержание

Список иллюстраций

Список таблиц

1 Цель работы

Получить навыки управления процессами операционной системы.

2 Выполнение

3 Выполнение

3.1 Управление заданиями

1. Для начала работы были получены полномочия администратора с помощью команды **su -**.

2. Далее запущены процессы:

   - **sleep 3600 &** — задание 1 (ожидание в течение часа в фоновом режиме);

   - **dd if=/dev/zero of=/dev/null &** — задание 2 (копирование потока нулей в `/dev/null` в фоне);

   - **sleep 7200** — задание 3 (ожидание 2 часа в режиме переднего плана).

3. Поскольку третья команда была запущена без &, оболочка была заблокирована. С помощью комбинации **Ctrl+Z** процесс был остановлен (статус *Stopped*).

4. Для просмотра списка заданий выполнена команда **jobs**, отобразившая три задания: два выполнялись в фоновом режиме, одно находилось в состоянии *Stopped*.

5. Задание 3 было переведено в фоновый режим командой **bg 3**. Повторная проверка через **jobs** показала, что все процессы находятся в состоянии *Running*.

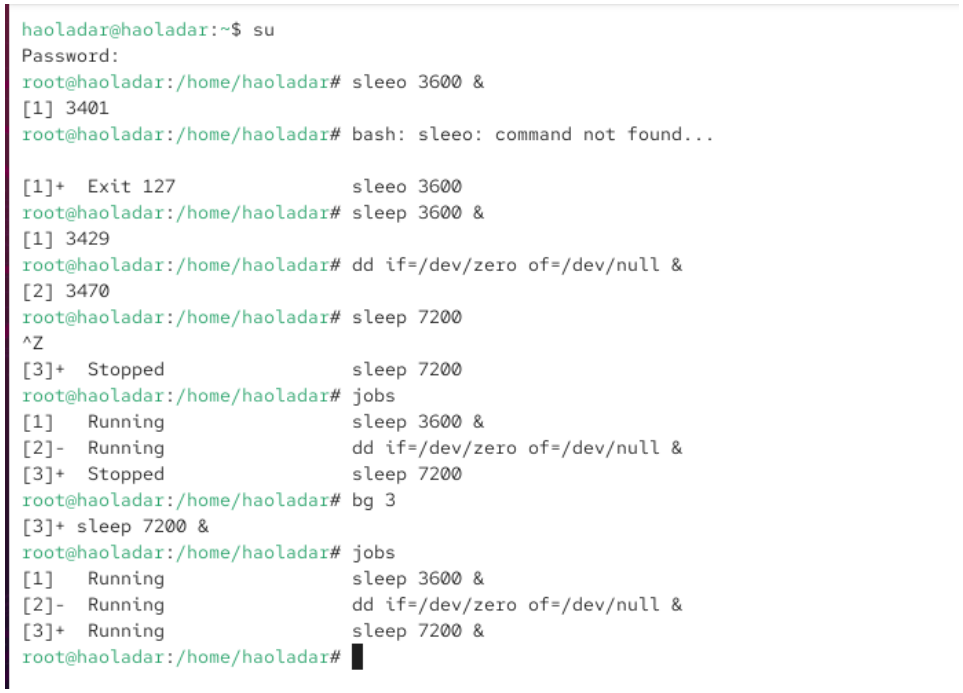

Рис. 3.1: Продолжение выполнения job в фоне

6. Для возврата задания 1 на передний план применена команда **fg 1**. Процесс был прерван комбинацией **Ctrl+C**. Аналогичным образом завершены задания 2 и 3.

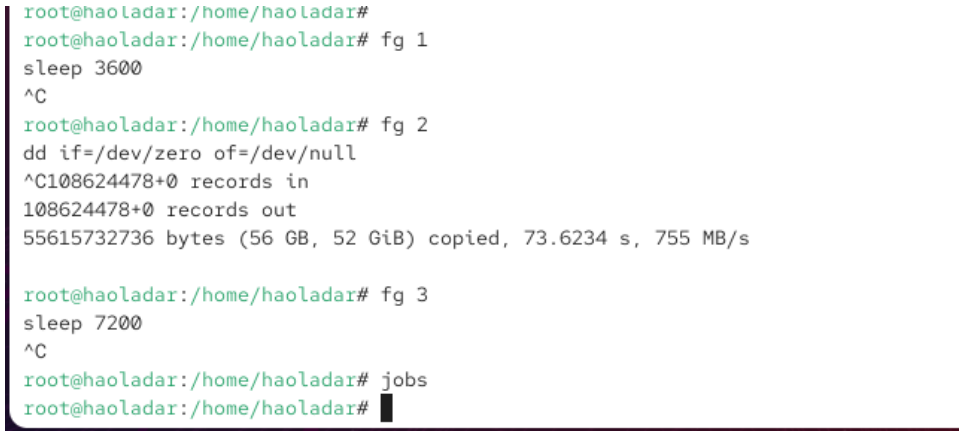

Рис. 3.2: Перевод процессов на передний план и завершение

7. В отдельном терминале под пользователем была запущена команда: **dd if=/dev/zero of=/dev/null &**

Процесс успешно стартовал и продолжил выполнение в фоне.

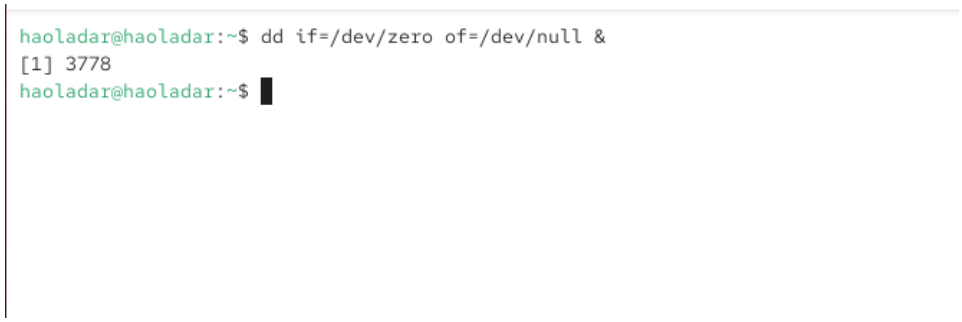

Рис. 3.3: Фоновый процесс в другом терминале

8. После закрытия терминала команда **top** показала, что процесс **dd** всё ещё работает от имени пользователя.

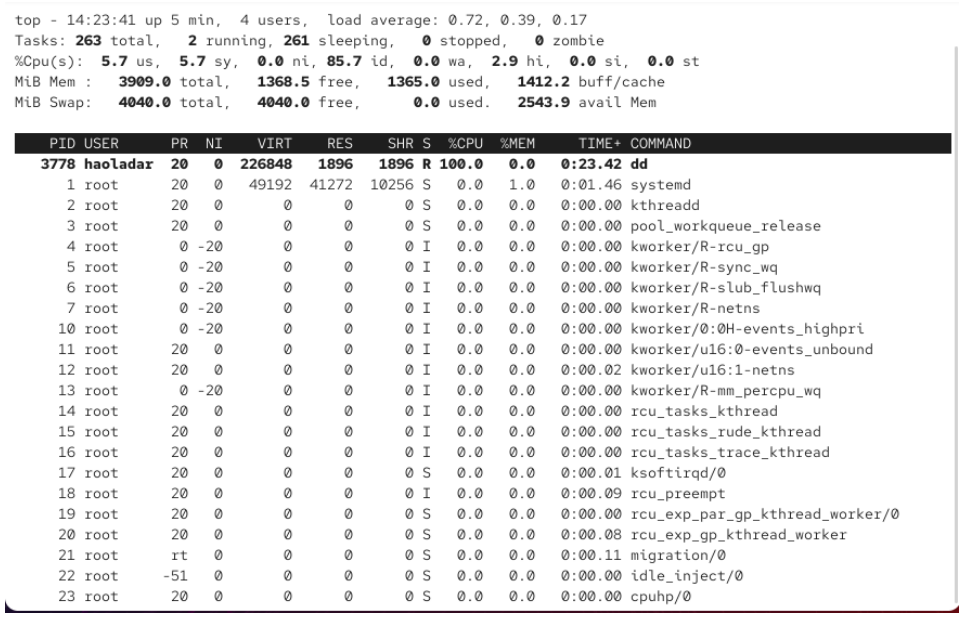

Рис. 3.4: Просмотр процессов через top

9. Для завершения работы процесса в **top** была использована команда **k**. После подтверждения PID процесс был завершён.

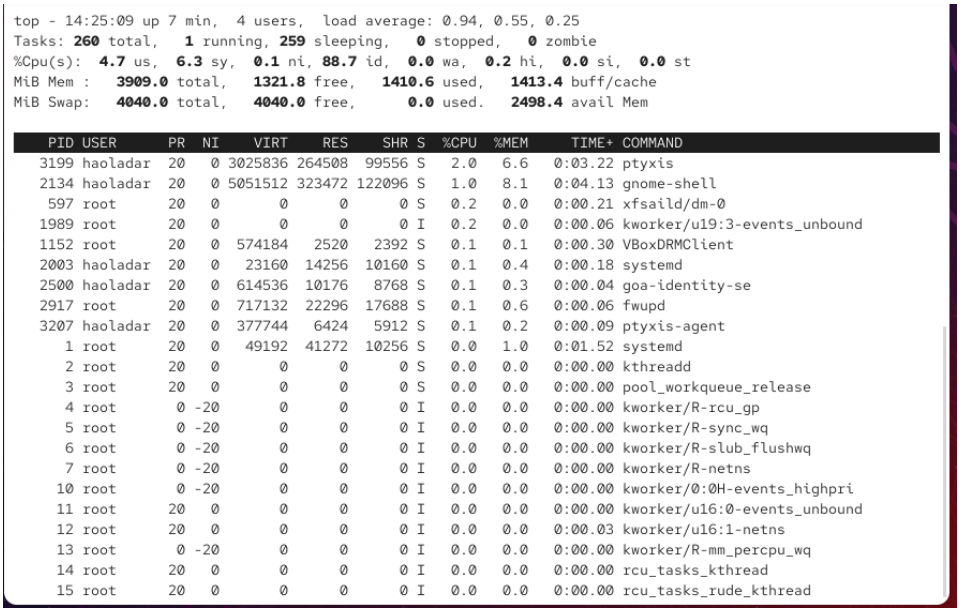

Рис. 3.5: Завершение процесса dd через top

3.2 Управление процессами

1. Для начала были получены полномочия администратора с помощью команды **su -**.

2. Запущены три фоновых процесса:

   - **dd if=/dev/zero of=/dev/null &** — процесс 1

   - **dd if=/dev/zero of=/dev/null &** — процесс 2

   - **dd if=/dev/zero of=/dev/null &** — процесс 3

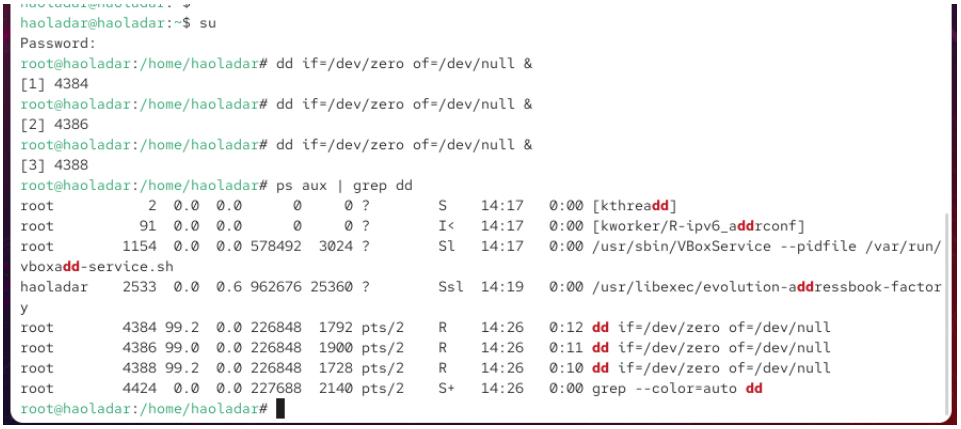

Рис. 3.6: Запуск фоновых процессов dd

3. Для проверки списка запущенных процессов использована команда **ps aux | grep dd**.

   В выводе показаны все строки, содержащие dd. Три последних строки соответствуют активным процессам dd, запущенным ранее.

4. С помощью команды **renice -n 5 <PID>** можно изменить приоритет выполнения одного из процессов dd, задав новое значение.

5. Для анализа иерархии процессов выполнена команда **ps fax | grep -B5 dd**.

   Параметр `-B5` добавил пять строк выше совпадения, что позволило отобразить дерево процессов и определить родительскую оболочку, из которой были запущены процессы dd.

6. Был определён PID оболочки, породившей процессы dd.

   Для завершения всех связанных процессов выполнена команда **kill -9 <PID>**.

   В результате оболочка завершила работу, а вместе с ней были остановлены и все дочерние процессы dd.

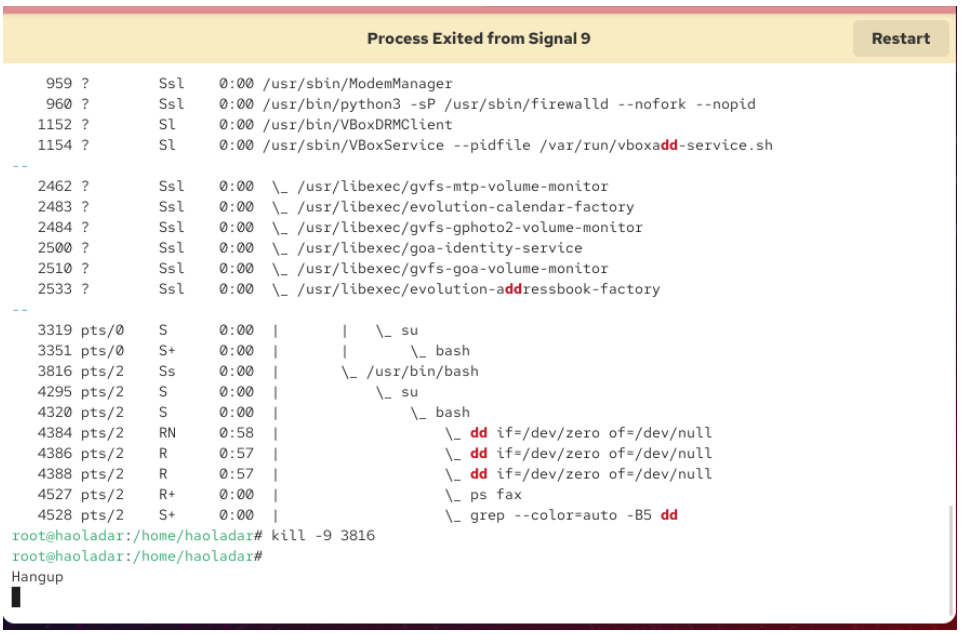

Рис. 3.7: Завершение процессов через kill

3.3 Задание 1

1. Были запущены три фоновых процесса с помощью команды:
   dd if=/dev/zero of=/dev/null &

   В результате для каждого процесса был назначен собственный PID.

2. Приоритет одного из процессов был изменён с помощью команды **renice -n -5 <PID>**.
   Это позволило увеличить его приоритет (чем меньше значение, тем выше приоритет).

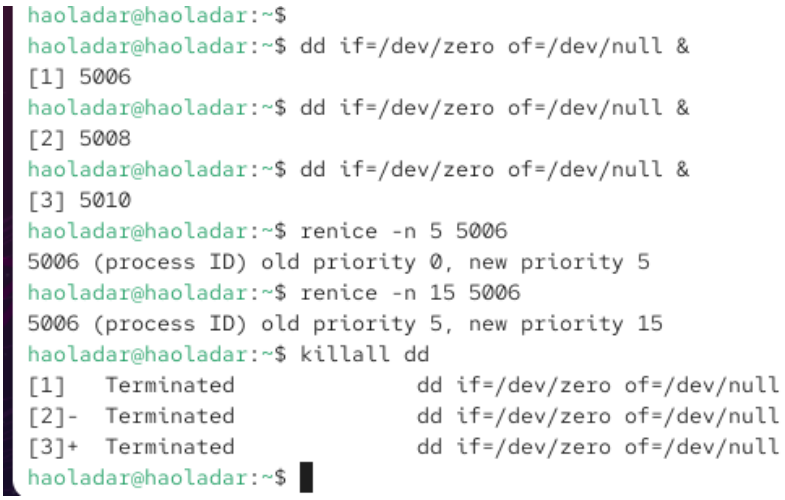

Рис. 3.8: Изменение приоритета процесса

3. Приоритет того же процесса был изменён повторно с помощью команды **renice -n 15 <PID>**, что понизило его приоритет по сравнению с другими.

4. Для завершения всех процессов dd использована команда **killall dd**.
   Все три фоновых процесса были корректно завершены.

3.4 Задание 2

1. Запущена программа **yes** в фоновом режиме с подавлением потока вывода: **yes > /dev/null &**

2. Программа **yes** была запущена на переднем плане с перенаправлением вывода.
   Сначала её выполнение приостановлено комбинацией **Ctrl+Z**, затем возобновлено, и процесс был завершён с помощью **Ctrl+C**.

Рис. 3.9: Работа yes на переднем плане и завершение

3. Аналогичные действия были выполнены с программой **yes** без перенаправления вывода: приостановка (Ctrl+Z), возобновление и завершение (Ctrl+C).

4. Для проверки состояний процессов использована команда **jobs**. Отображены как выполняющиеся (*Running*), так и остановленные (*Stopped*) задания.

5. Один из фоновых процессов был выведен на передний план с помощью **fg <номер>** и завершён.

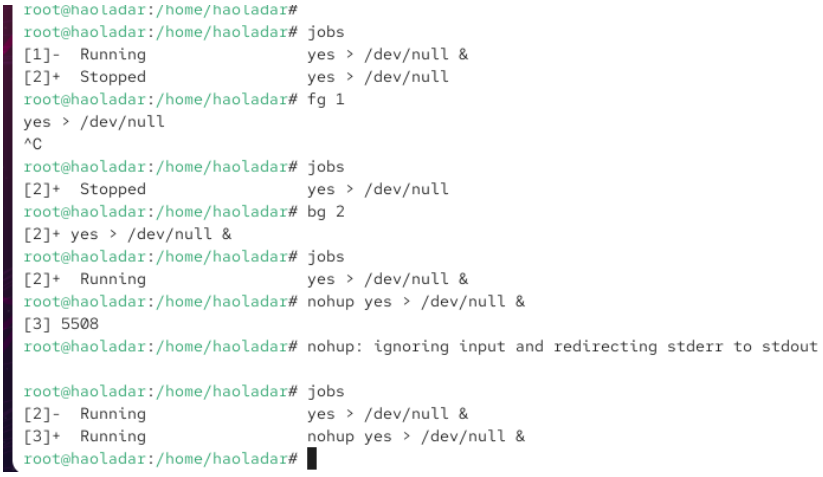

Рис. 3.10: Перевод процесса на передний план

6. Остановленный процесс был возвращён в фоновый режим командой **bg <номер>**. Его статус изменился на *Running*.

7. Для запуска процесса, который продолжит работу после выхода из терминала, использована команда:

   nohup yes > /dev/null &

   Это позволило игнорировать сигнал SIGHUP.

8. С помощью команды **top** просмотрен список процессов. В нём видно, что **yes** активно загружает CPU.

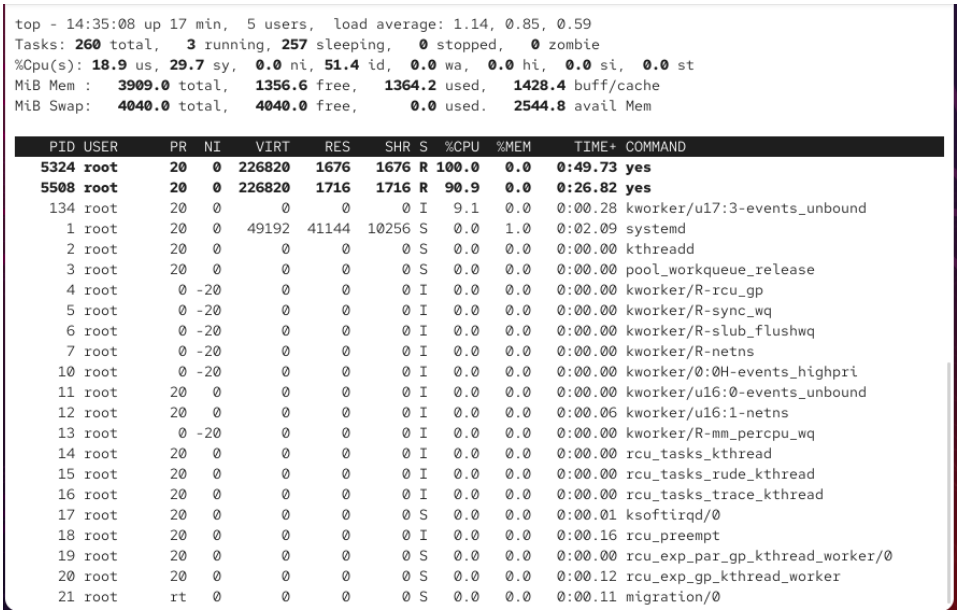

Рис. 3.11: Вывод top с процессами yes

9. Запущено ещё несколько процессов **yes** в фоновом режиме. Для завершения части из них использованы команды:

   - **kill <PID>** — завершение по PID

   - **kill %<номер>** — завершение по идентификатору задания

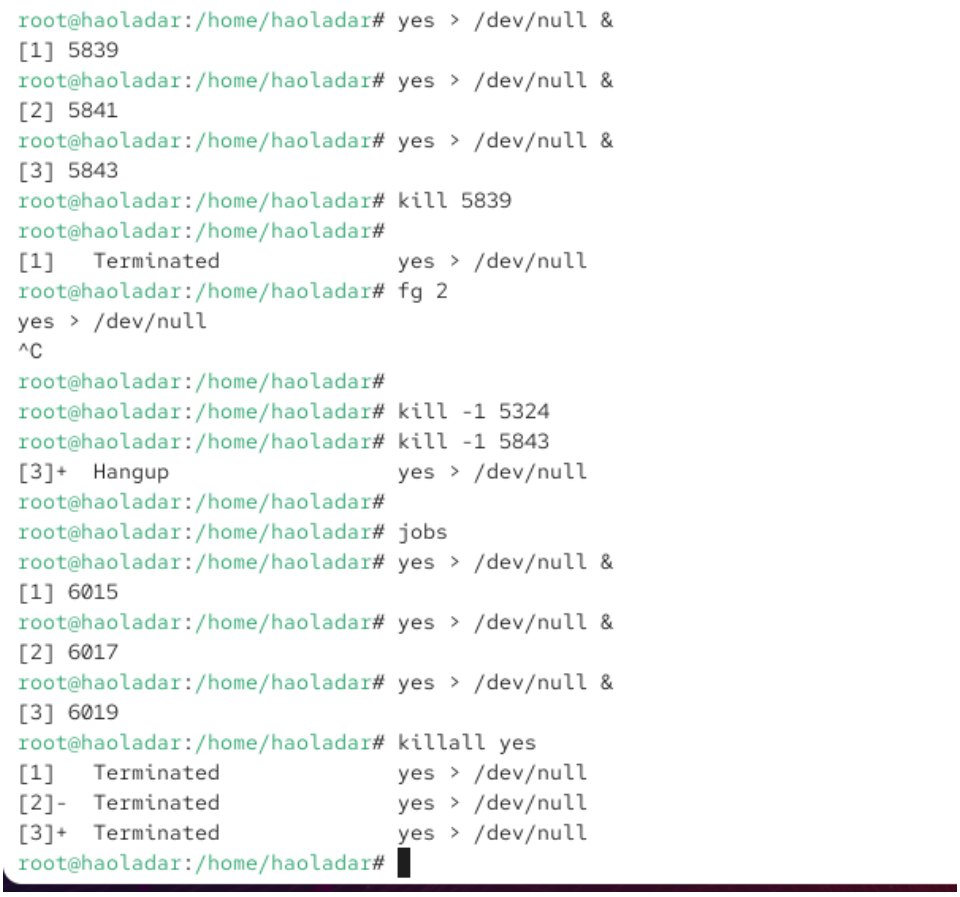

Рис. 3.12: Завершение процессов по PID и job ID

10. Для проверки обработки сигналов был отправлен сигнал **SIGHUP (1)** обычному процессу и процессу, запущенному через **nohup**.
    Первый завершился, второй продолжил работу.

11. Запущено несколько новых процессов **yes**, которые затем были завершены одной командой:
    killall yes

12. Для сравнения приоритетов два процесса были запущены так:

    - **yes > /dev/null &** (обычный приоритет)

    - **nice -n 5 yes > /dev/null &** (с приоритетом +5).
      Проверка через **ps -l** показала различие в приоритетах (*NI*).

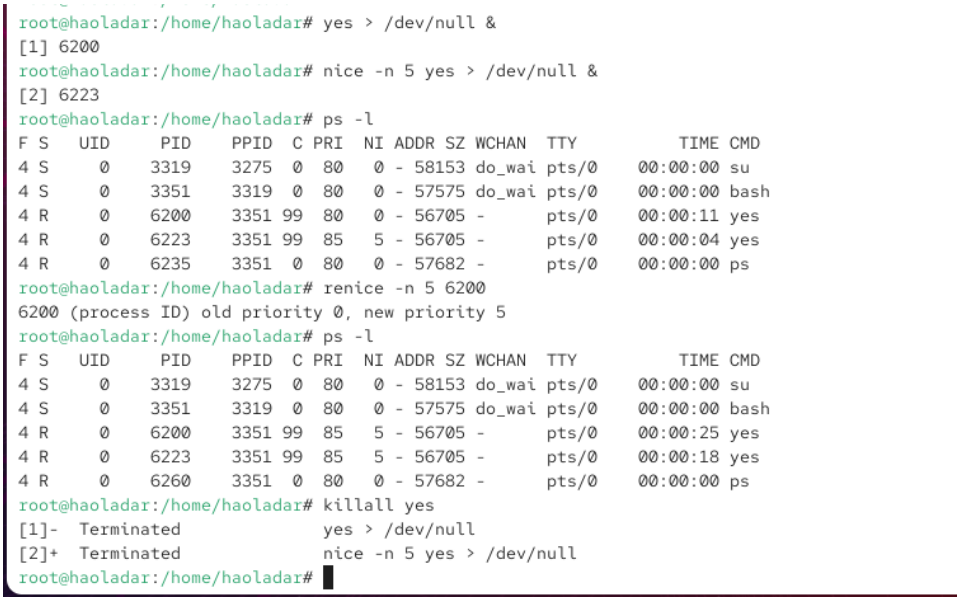

Рис. 3.13: Сравнение приоритетов процессов

13. Один из процессов был изменён с помощью **renice**, чтобы выровнять приоритеты обоих потоков.

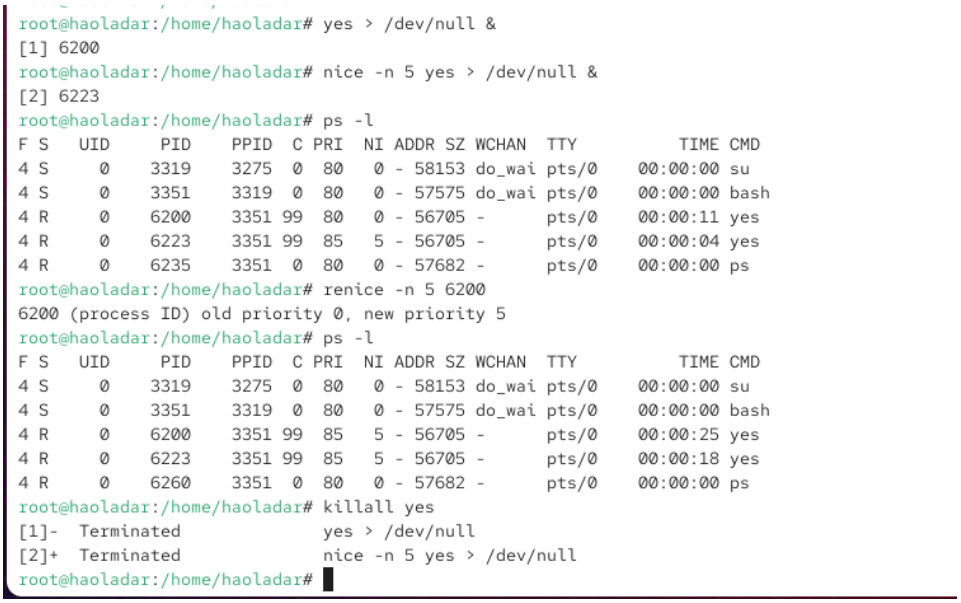

Рис. 3.14: Изменение приоритетов с помощью renice

4 Контрольные вопросы

1. Какая команда позволяет вам искать пакет rpm, содержащий файл useradd?
   rpm -qf /usr/sbin/useradd — если файл уже установлен.
   dnf provides *useradd или repoquery -f* /useradd — если нужно найти пакет, в который входит этот файл.

2. Какие команды вам нужно использовать, чтобы показать имя группы dnf, которая содержит инструменты безопасности и показать, что находится в этой группе?

   - **dnf group list** — список всех групп.

   - **dnf group info “Security Tools”** — подробности о содержимом группы.

3. Какая команда позволяет вам установить rpm, который вы загрузили из Интернета и который не находится в репозиториях?
   rpm -Uhv имя_пакета.rpm
   (или с проверкой зависимостей: **dnf install ./имя_пакета.rpm**).

4. Вы хотите убедиться, что пакет rpm, который вы загрузили, не содержит никакого опасного кода сценария. Какая команда позволяет это сделать?
   rpm -qp –scripts имя_пакета.rpm — показывает скрипты, которые выполняются при установке/удалении.

5. Какая команда показывает всю документацию в rpm?
   rpm -qd имя_пакета

6. Какая команда показывает, какому пакету rpm принадлежит файл?
   rpm -qf /путь/к/файлу

5 Заключение

Освоены базовые навыки работы с пакетным менеджером **dnf** и утилитой **rpm** для установки, поиска, изучения и удаления пакетов и групп в Linux.