Липецкий государственный технический университет

Факультет автоматизации и информатики

Кафедра автоматизированных систем управления

ЛАБОРАТОРНАЯ РАБОТА №2

по дисциплине «Операционная система Linux»

Процессы в операционной системе Linux

Студент Осипов А.А.

Группа АИ-18

Руководитель Кургасов В.В.

Липецк 2020 г.

Оглавление

Цель работы

Ознакомиться на практике с понятием процесса в операционной системе. Приобрести опыт и навыки управления процессами в операционной системе Linux.

Задание кафедры

Часть I

1) Загрузиться не root, а пользователем.

2) Найти файл с образом ядра. Выяснить по имени файла номер версии Linux.

3) Посмотреть процессы ps –f. Прокомментировать. Для этого почитать man ps.

4) Написать с помощью редактора vi два сценария loop и loop2. Текст сценариев:

Loop:

while true; do true; done

Loop2:

while true; do true; echo ‘Hello’; done

5) Запустить loop2 на переднем плане: sh loop2.

6) Остановить, послав сигнал STOP.

7) Посмотреть последовательно несколько раз ps –f. Записать сообщение, объяснить.

8) Убить процесс loop2, послав сигнал kill -9 PID. Записать сообщение.

Прокомментировать.

9) Запустить в фоне процесс loop: sh loop&. Не останавливая, посмотреть несколько раз: ps –f. Записать значение, объяснить.

10) Завершить процесс loop командой kill -15 PID. Записать сообщение,

прокомментировать.

11) Третий раз запустить в фоне. Не останавливая убить командой kill -9 PID.

12) Запустить еще один экземпляр оболочки: bash.

13) Запустить несколько процессов в фоне. Останавливать их и снова запускать. Записать результаты просмотра командой ps –f.

Часть II

1. Запустить в консоли на выполнение три задачи, две в интерактивном

режиме, одну - в фоновом.

2. Перевести одну из задач, выполняющихся в интерактивном режиме, в фоновый режим.

3. Провести эксперименты по переводу задач из фонового режима в интерактивный и наоборот.

4. Создать именованный канал для архивирования и осуществить передачу в канал

- списка файлов домашнего каталога вместе с подкаталогами (ключ -R);
- одного каталога вместе с файлами и подкаталогами.

5. В отчете предоставьте все шаги ваших действий. То есть следует привести следующее: текст задания, а следом за ним снимок экрана консоли с результатами выполнения задания. Кроме того, перед скриншотом следует привести текстовую запись использованных команд.

Часть III Индивидуальное задание

Вариант 2

1. Получить следующую информацию о процессах текущего пользователя: идентификатор и имя владельца процесса, статус и приоритет процесса.

2. Завершить выполнение двух процессов, владельцем которых является текущий пользователь. Первый процесс завершить с помощью сигнала SIGINT, задав его имя, второй — с помощью сигнала SIGQUIT, задав его номер.

3. Определить идентификаторы и имена процессов, идентификатор группы которых не равен идентификатору группы текущего пользователя.

4. В отчете предоставьте все шаги ваших действий. То есть следует привести следующее: текст задания, а следом за ним снимок экрана консоли с результатами выполнения задания. Кроме того, перед скриншотом следует привести текстовую запись использованных команд. Кратко поясните результаты выполнения всех команд.

Ход работы

Часть I

Запустим виртуальную машину Linux Ubuntu и загрузимся не root, а пользователем. Найдём файл с обзором ядра и выясним по имени файла номер версии Linux. Результаты работы представлены на рисунке 1.

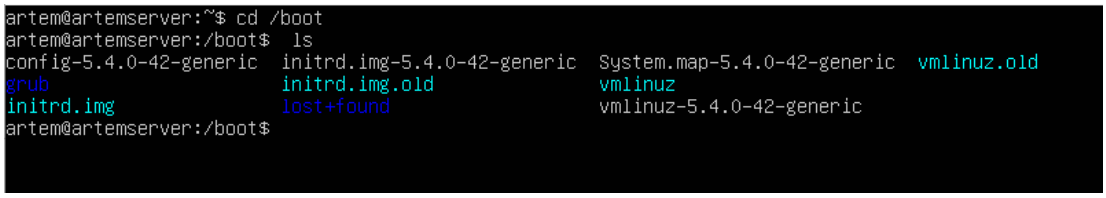

Рисунок 1 – Загрузка пользователем, версия ядра

Исходя из рисунка 1 ядро имеет версию 5.4.0

Далее на рисунке 2 посмотрим процессы ps –f.

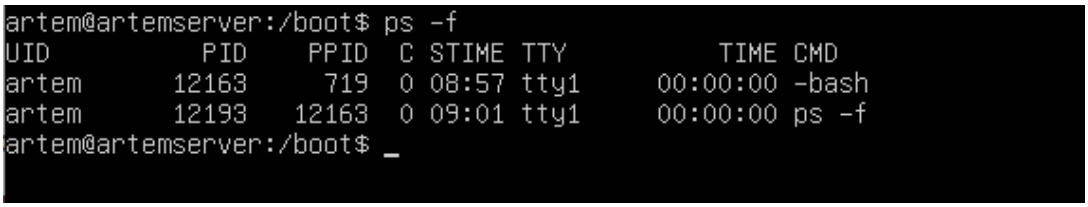

Рисунок 2 – Просмотр процессов ps -f

Поясним просмотр процессов.

UID - имя пользователя, от имени которого работает процесс;

PID - идентификатор пользователя;

PPID - идентификатор родительского процесса пользователя;

C - расходование ресурсов процессора, в процентах;

STIME - время, когда процесс был запущен;

TTY - если процесс привязан к терминалу, то здесь будет выведен его номер;

TIME - общее время выполнения процесса (user + system);

CMD - команда, которой был запущен процесс, если программа не может прочитать аргументы процесса, он будет выведен в квадратных скобках;

Напишем с помощью редактора vi два сценария loop и loop2. Текст сценариев: (cd /home). Результаты работы приведены на рисунках 3, 4 и 5.

Loop:

while true; do true; done

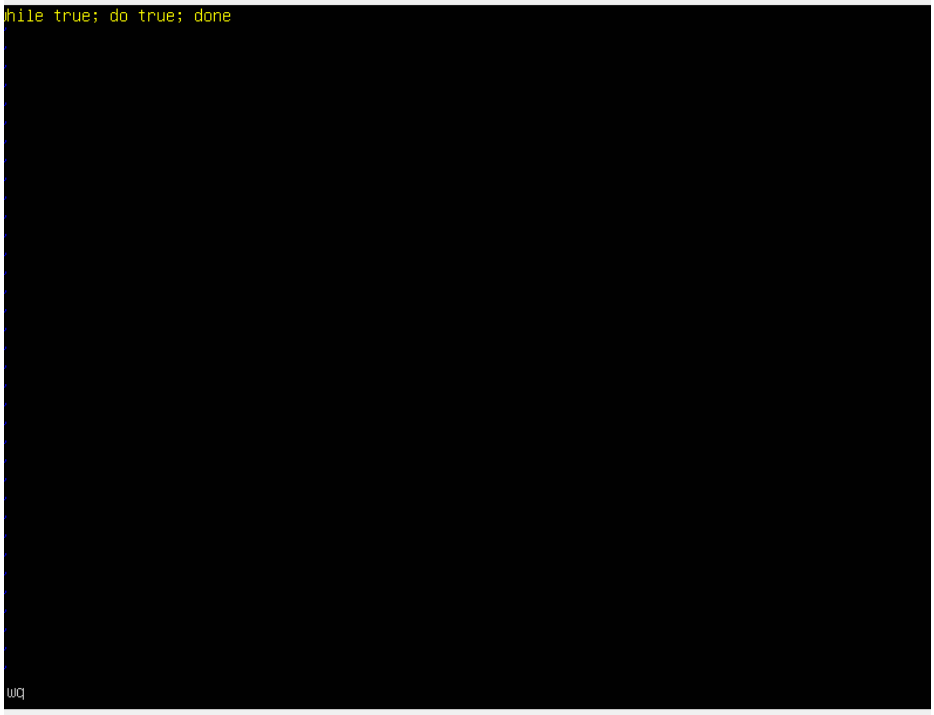

Рисунок – 3 Написание сценария loop

Рисунок 4 – Написание сценария loop2

Loop2:

while true; do true; echo ‘Hello’; done

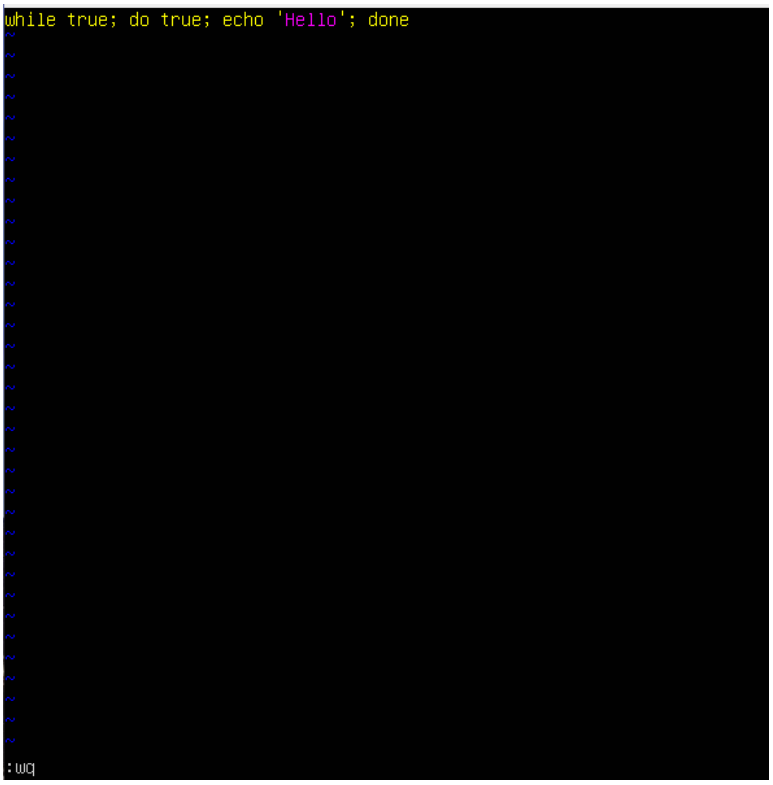

Рисунок 5 – Написание сценария loop2

Следующим шагом приведём пример каталога на наличие loop.sh и loop2.sh. Результат работы представлен на рисунке 6.

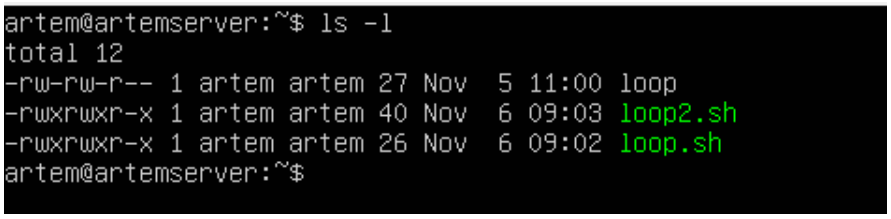

Рисунок 6 – Пример каталога с файлами

Далее запустим loop2 на переднем плане: sh loop2.sh и остановим, послав сигнал STOP. Результат работы представлен на рисунке 7.

Рисунок 7 – Запуск loop 2 на переднем плане loop2.sh и его остановка сигналом стоп

Следующим заданием посмотрим последовательно несколько раз ps –f. Результат работы представлен на рисунке 8.

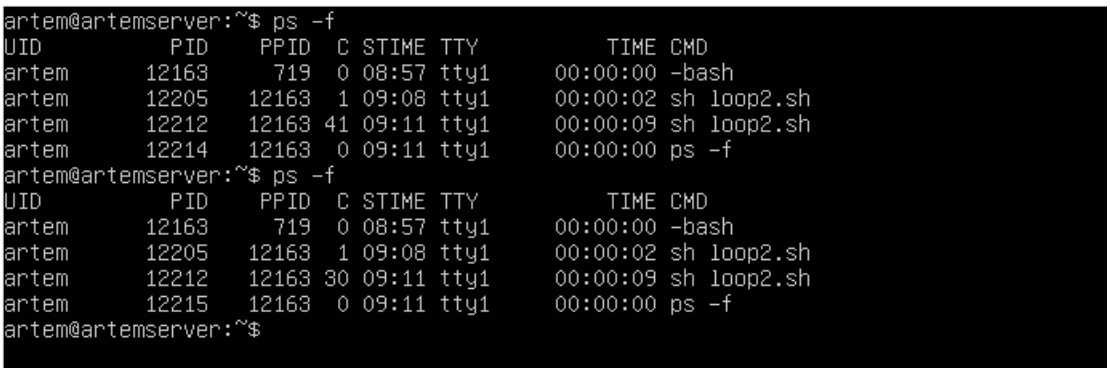

Рисунок 8 – Просмотр листинга процессов

Из рисунка 8 следует, что расходование ресурсов процессора, в процентах, затраченный на процесс loop2, уменьшается, что говорит о приостановке процессора.

Далее нужно убить процесс loop2, послав сигнал kill -9 PID. Результат работы представлен на рисунке 9.

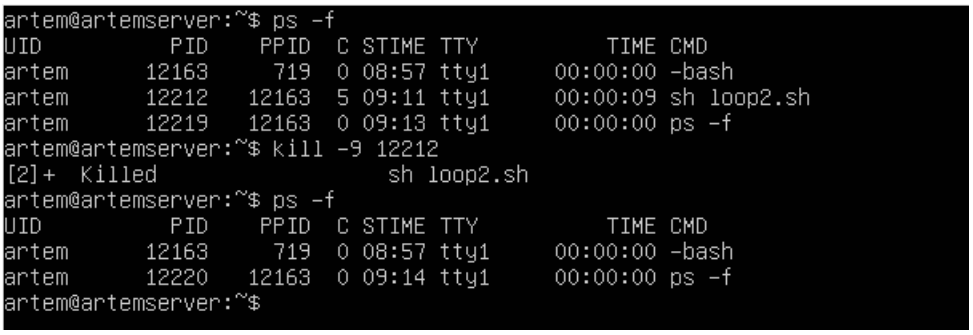

Рисунок 9 – Уничтожение процесса loop2.sh

Далее нужно запустить в фоне процесс loop: sh loop&. Не останавливая, посмотреть несколько раз: ps –f. Результат работы представлен на рисунке 10.

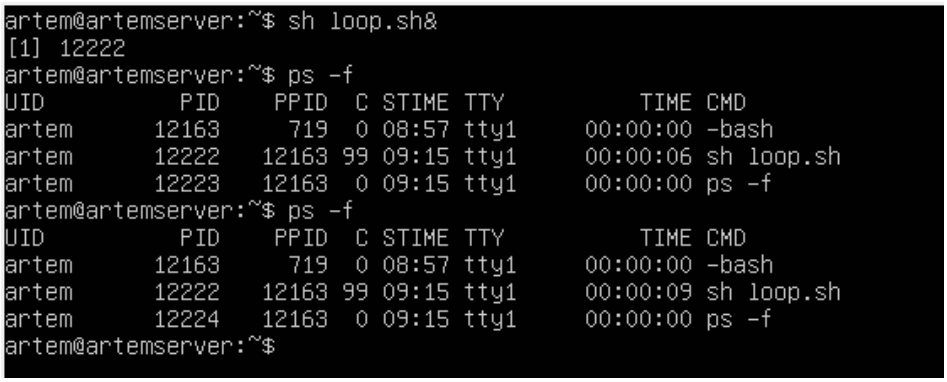

Рисунок 10 – Запуск процесса в фоне и просмотр ps -f

По рисунку 10 заметно, что расходование ресурсов процессора, в процентах, затраченный на процесс loop.sh, не уменьшается, что говорит о том что процесс запущен.

Далее завершим процесс loop командой kill -15 PID. Результат работы представлен на рисунке 11.

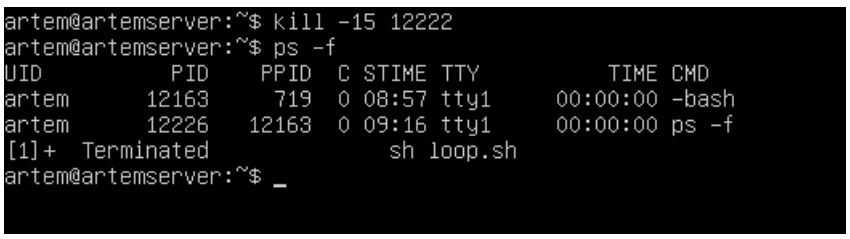

Рисунок 11 – Заверение процесса loop.sh

А теперь снова запустим в фоне процесс loop.sh и не останавливая убьём командой kill -9 PID. Результат работы представлен на рисунке 12.

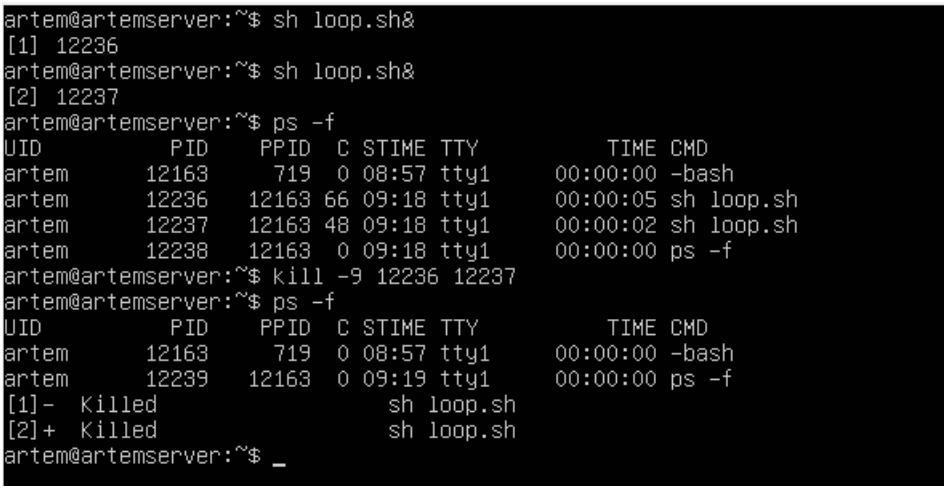

Рисунок 12 – Запуск процесса loop.sh в фоне и его убийство командой kill -9

Запустим еще один экземпляр оболочки с помощью команды bash. Результат работы представлен на рисунке 13.

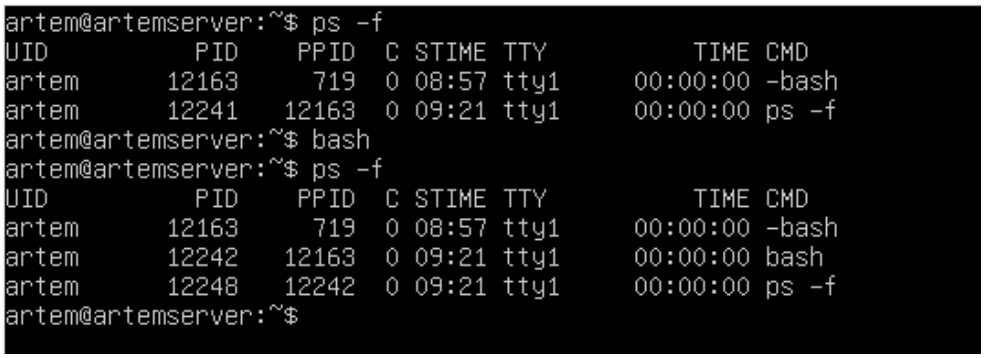

Рисунок 13 – Запуск еще одного экземпляра

Запустим несколько процессов в фоне. Будем останавливать их и снова запускать. Результат работы представлен на рисунке 14.

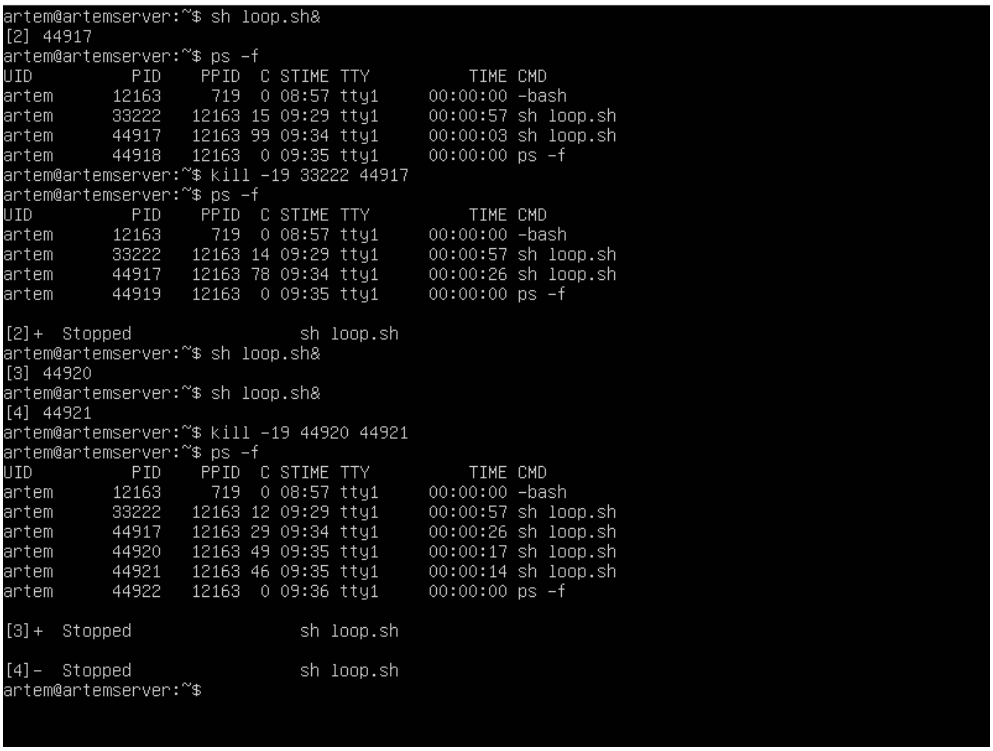

Рисунок 14 – Запуск процессов в фоне и их остановка

Часть II

Нужно запустить в консоли на выполнение три задачи, две в интерактивном режиме, одну - в фоновом (sh loop.sh, sh loop.sh, sh loop.sh&). Далее посмотрим все процессы с помощью командой jobs. Результат работы представлен на рисунке 15.

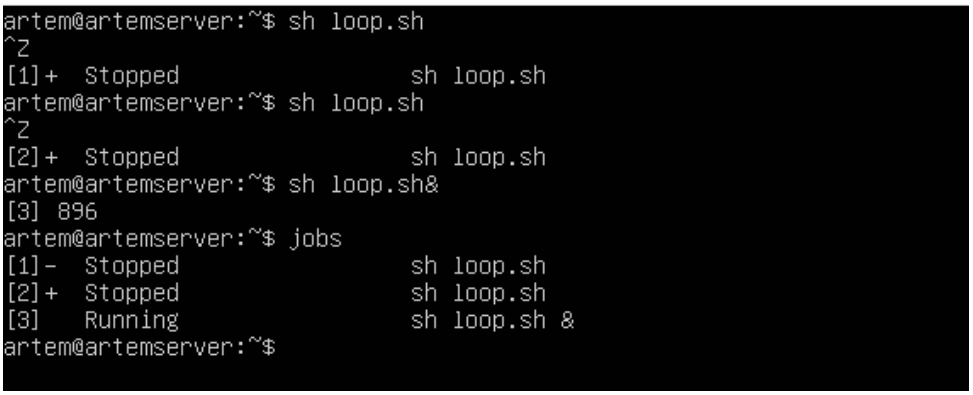

Рисунок 15 – Запуск на выполнение трёх задач

Следующим заданием нужно перевести одну из задач, выполняющихся в интерактивном режиме, в фоновый режим (bg). Результат работы представлен на рисунке 16.

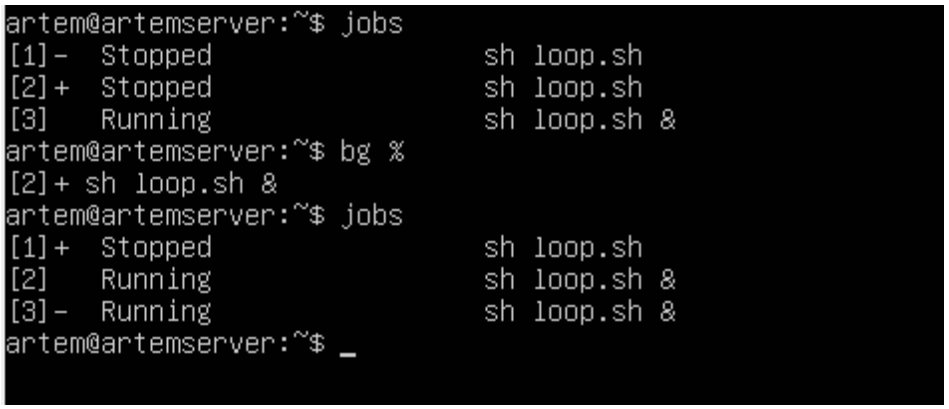

Рисунок 16 – Перевод задачи выполняющейся в интерактивном режиме, в фоновый режим

Далее проведём эксперименты по переводу задач из фонового режима в интерактивный и наоборот. Результат работы представлен на рисунке 17.

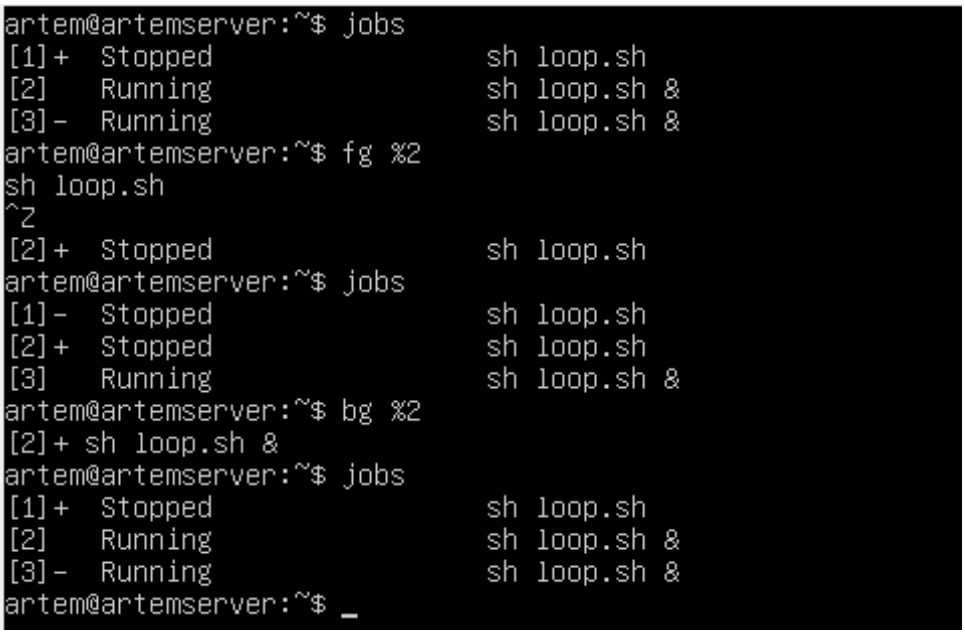

Рисунок 17 – Эксперименты по переводу задач

Нужно создать именованный канал для архивирования и осуществить передачу в канал списка файлов домашнего каталога вместе с подкаталогами (ключ -R), одного каталога вместе с файлами и подкаталогами. Результат работы представлен на рисунках 18, 19, 20, 21 и 22.

```
artem@artemserver:~$ ls -l
total 12
-rw-rw-r-- 1 artem artem 27 Nov  5 11:00 loop
-rwxrwxr-x 1 artem artem 40 Nov  6 09:03 loop2.sh
-rwxrwxr-x 1 artem artem 26 Nov  6 09:02 loop.sh
artem@artemserver:~$ mkfifo channel
artem@artemserver:~$ ls -l
total 12
prw-rw-r-- 1 artem artem  0 Nov  6 09:49 channel
-rw-rw-r-- 1 artem artem 27 Nov  5 11:00 loop
-rwxrwxr-x 1 artem artem 40 Nov  6 09:03 loop2.sh
-rwxrwxr-x 1 artem artem 26 Nov  6 09:02 loop.sh
artem@artemserver:~$
```

Рисунок 18 – Создание именованного канала

```
artem@artemserver:~$ ls -l
total 12
prw-rw-r-- 1 artem artem  0 Nov  6 09:49 channel
-rw-rw-r-- 1 artem artem 27 Nov  5 11:00 loop
-rwxrwxr-x 1 artem artem 40 Nov  6 09:03 loop2.sh
-rwxrwxr-x 1 artem artem 26 Nov  6 09:02 loop.sh
artem@artemserver:~$ ls -R > channel&
[4] 921
artem@artemserver:~$ _
```

Рисунок 19 – Передача в канал листинг домашнего каталога

```
artem@artemserver:~$ mkdir new
artem@artemserver:~$ cd new
artem@artemserver:~/new$ touch 1.txt
artem@artemserver:~/new$ touch 2.txt
artem@artemserver:~/new$ mkdir new1
artem@artemserver:~/new$ cd new1
artem@artemserver:~/new/new1$ touch 3.txt
artem@artemserver:~/new/new1$ cd ..
artem@artemserver:~/new$ ls -l
total 4
-rw-rw-r-- 1 artem artem    0 Nov  6 10:01 1.txt
-rw-rw-r-- 1 artem artem    0 Nov  6 10:01 2.txt
drwxrwxr-x 2 artem artem 4096 Nov  6 10:02 new1
artem@artemserver:~/new$ _
```

Рисунок 20 – Создание нового каталога

Рисунок 21 – Листинг домашнего каталога

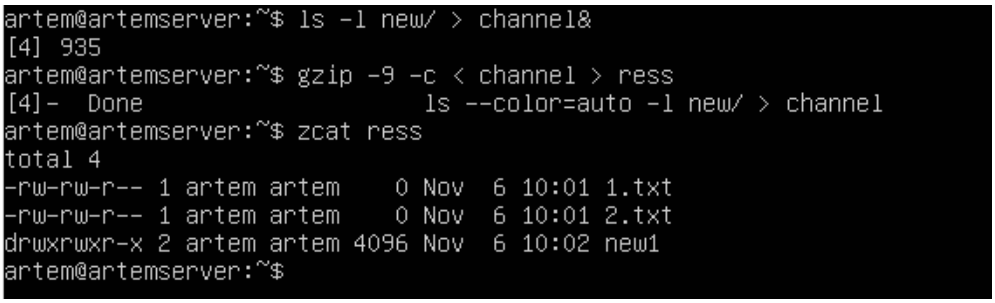

Рисунок 22 – Занесение в канал листинг созданного каталога и архивация содержимого каталога

Часть 3

Нужно получить следующую информацию о процессах текущего пользователя: идентификатор и имя владельца процесса, статус и приоретет процесса.

Для этого используем команду top - позволяет выводить информацию о системе, а также список процессов динамически обновляя информацию о потребляемых ими ресурсах.

-v - вывести версию программы;

-b - режим только для вывода данных, программа не воспринимает интерактивных команд и выполняется пока не будет завершена вручную;

-c - отображать полный путь к исполняемым файлам команд;

-d - интервал обновления информации;

-H - включает вывод потоков процессов;

-i - не отображать процессы, которые не используют ресурсы процессора;

-n - количество циклов обновления данных, после которых надо закрыть программу;

-o - поле, по которому надо выполнять сортировку;

-O - вывести все доступные поля для сортировки;

-p - отслеживать только указанные по PID процессы, можно указать несколько PID;

-u - выводить только процессы, запущенные от имени указанного пользователя.

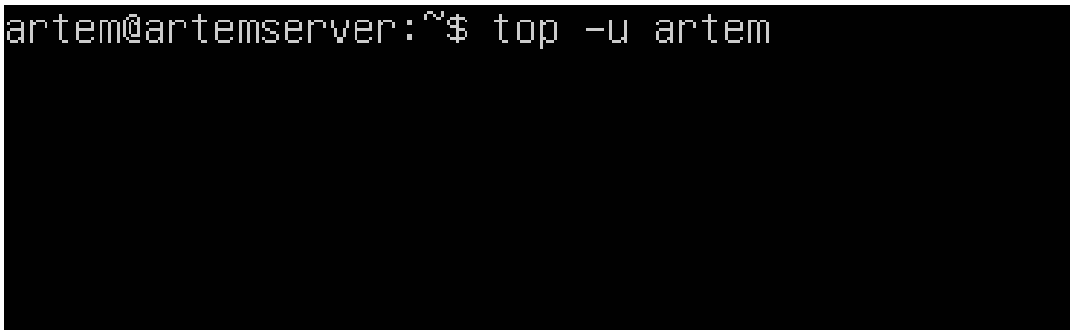

Рисунок 23 – использование команды top

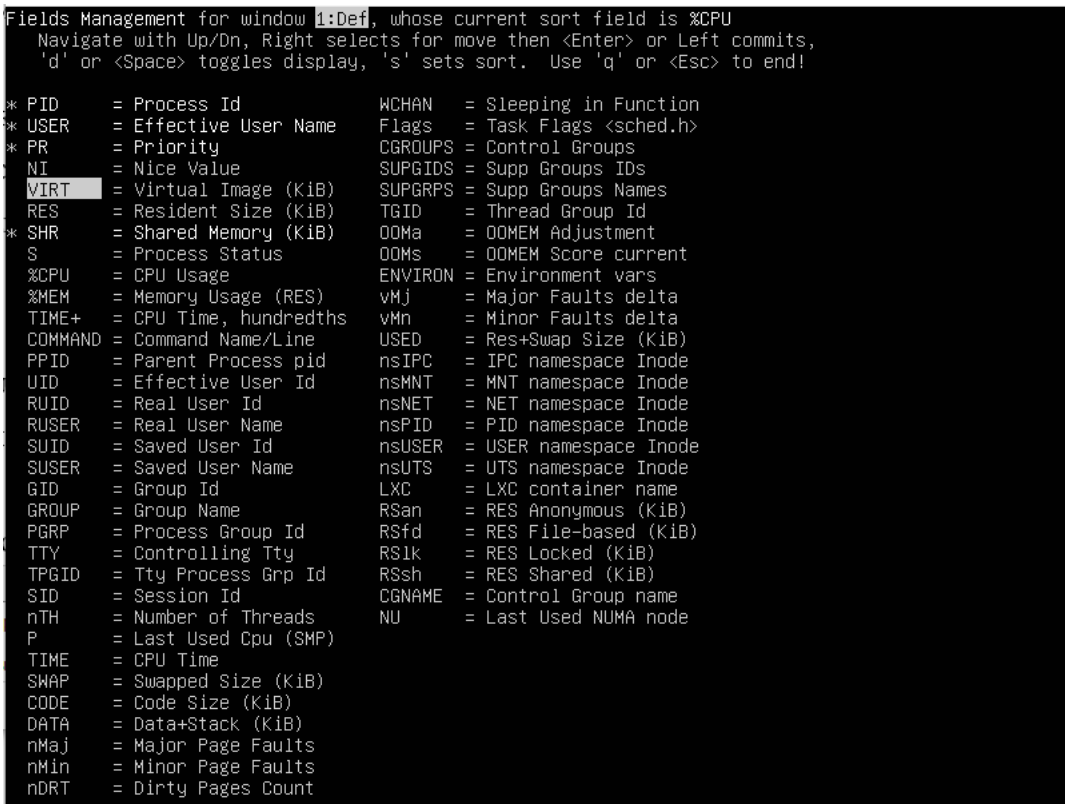

Рисунок 24 – настройка параметров вывода

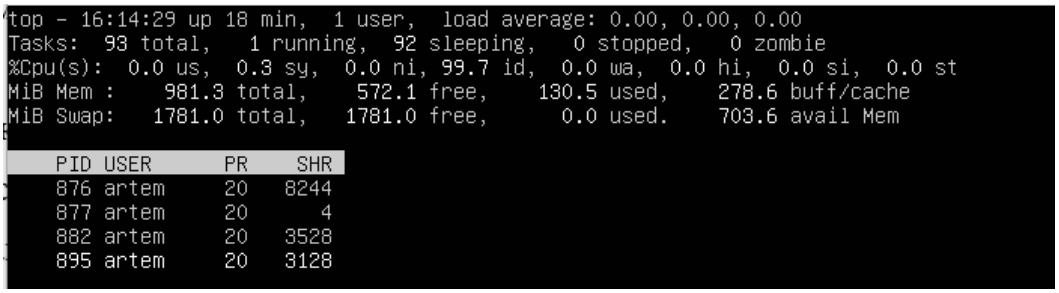

Рисунок 25 - получение информации о процессах текущего пользователя

Далее нужно завершить выполнение двух процессов, владельцем которых является текущий пользователь. Первый процесс завершить с помощью команды kill -2 (сигнал SIGINT), второй – с помощью сигнала SIGQUIT, задав его номер.

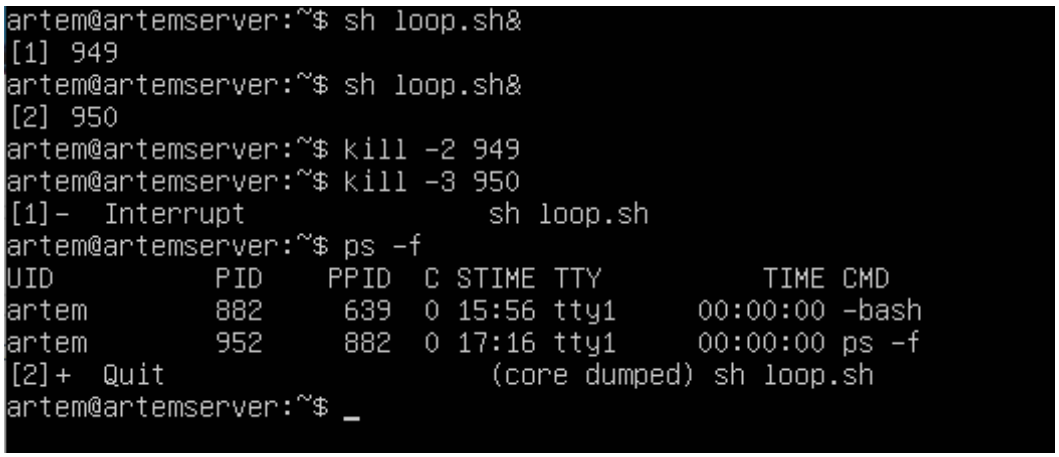

Рисунок 26 – остановка процессов

Определим идентификаторы и имена процессов, идентификатор группы которых не равен идентификатору группы текущего пользователя.

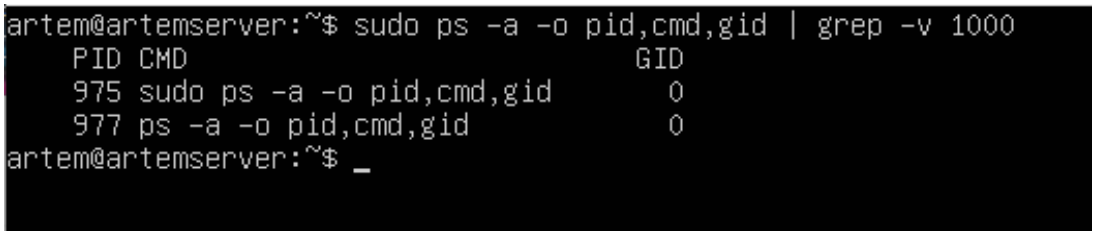

Рисунок 27 - Идентификаторы и имена процессов

Вывод

В ходе выполнения лабораторной работы я ознакомился на практике с понятием процесса в операционной системе, приобрел опыт и навыки управления процессами в операционной системе Linux.