Липецкий государственный технический университет

Кафедра автоматизированных систем управления

Полное и сокращенное название кафедры

Лабораторная работа № 2

по «Операционной системе Linux»

Наименование дисциплины

http://edu.kurgasov.ru/theme/image.php/clean/core/1486051463/spacer

Понятие о процессах в ОС Linux

Наименование темы

Студент Грунау Г.Ю.

фамилия, инициалы

Группа АИ-18-1

Руководитель

подпись Кургасов В.В.

Учетная степень, учетное звание фамилия, инициалы

Липецк 2020 г.

Содержание

[Цель работы 3](#_Toc54997879)

[Задание 4](#_Toc54997880)

[Ход работы 6](#_Toc54997881)

[1. Часть I. 6](#_Toc54997882)

[2. Часть II. 12](#_Toc54997883)

[3. Часть III. 16](#_Toc54997884)

[Вывод 21](#_Toc54997885)

# Цель работы

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

# Задание

Часть I.

1. Загрузиться не root, а пользователем.
2. Найти файл c 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.

1. 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.

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

1. Часть I.
   1. Загрузка пользователем, а не root



Рисунок 1 – Загрузка пользователем

* 1. Поиск файла с образом ядра

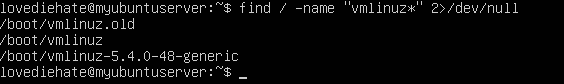


Рисунок 2 – Поиск образа ядра

На Рисунке 2 видно версию ядра – 5.4.0-48-generic.

* 1. Просмотр процессов ps –f

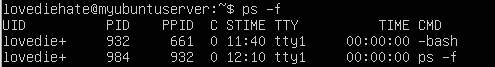


Рисунок 3 – Информация о процессах

На Рисунке 3 изображено использование команды ps -f. Параметр -f означает показ полного списка процессов.

* UID – идентификатор юзера, при параметре -f показывается входное его имя.
* PID – идентификатор процесса.
* PPID – идентификатор родительского процесса.
* C – доля выделенного планировщиком времени процессора.
* STIME – время или дата запуска процесса.
* TTY – управляющий терминал.
* TIME – истраченное процессом время процессора.
* CMD – имя программы и её аргументы.
  1. Написание двух сценариев с помощью редактора vi



Рисунок 4 – Первый сценарий loop



Рисунок 5 – Второй сценарий loop2

* 1. Запуск loop2 на переднем плане

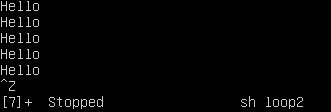


Рисунок 6 – Запуск и остановка процесса

* 1. Остановка процесса сигналом STOP

На Рисунке 6 изображена остановка активного процесса сигналом STOP нажатием клавиш Ctrl+Z.

* 1. Последовательный запуск ps –f

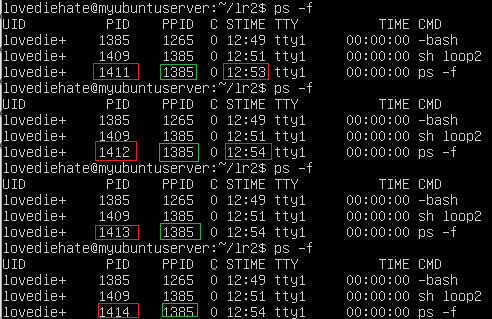


Рисунок 7 – Поочередный запуск ps –f

На Рисунке 7 видно, что при последовательном запуске программы идентификатор процесса и время запуска меняется. Это может говорить о том, что каждый раз запускается новый процесс, которому присваивается новый свободный идентификатор. При этом, родительский идентификатор не меняется: все процессы вызываются из одного интерпретатора –bash.

* 1. Убийство процесс loop2



Рисунок 8 – Убийство

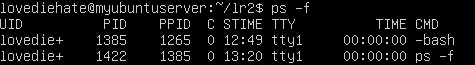


Рисунок 9 – Просмотр процессов

На Рисунке 8 с помощью команды kill -9 1409 был послан сигнал KILL процессу 1409. Этот сигнал означает немедленное принудительное завершение процесса без возможности перехвата этого сигнала.

На Рисунке 9 видим, что процесса с PID 1409 больше нет.

* 1. Запуск на фоне loop

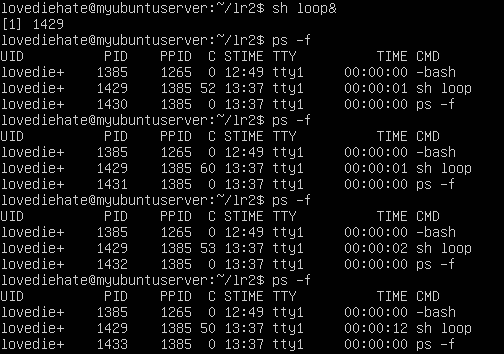


Рисунок 10 – Запуск фонового процесса

На Рисунке 10 заметно изменение значений в колонке C и TIME. Изменение в колонке C означает, что меняется % приоритета CPU, используемого процессом. TIME показывает, как долго запущен процесс. Последний вызов ps -f был через 12 секунд после запуска процесса 1429.

* 1. Завершение процесса командой kill -15

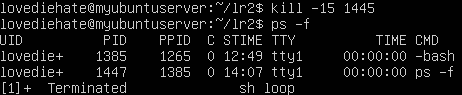


Рисунок 11 – Завершение процесса

На Рисунке 11 видно, что процесс завершён. Также в нижней строке мы видим Terminated sh loop, что может говорить нам о корректном завершении процесса.

* 1. Запуск в фоновом режиме и убийство процесса

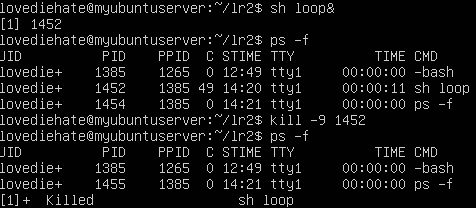


Рисунок 12 – Запуск и убийство

На Рисунке 12 видно, как файл loop запускается фоном и убивается командой kill -9. При первом просмотре моментальной съёмки статуса процессов выводится сообщение о том, что процесс успешно убит (см. последнюю строку на Рисунке 12).

* 1. Запуск оболочки bash

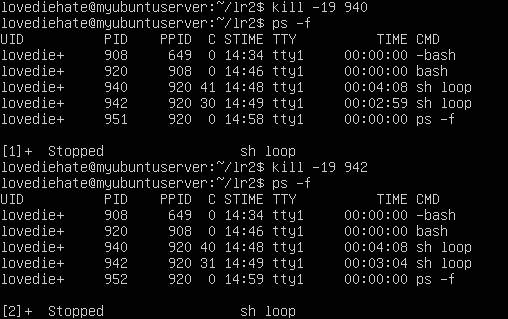


Рисунок 13 – Снимок процессов

На Рисунке 13 видно, что запущена оболочка bash с PID 920. Она будет родительским процессом последующих вызываемых процессов.

* 1. Запуск и остановка нескольких процессов

На Рисунке 13 видно, что запущено два процесса sh loop, и при остановке этих процессов выводится следующее сообщение: [1]+ Stopped sh loop – остановлен первый процесс sh loop.

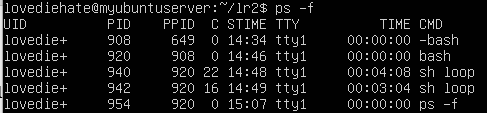


Рисунок 14 – Снимок процессов

На Рисунке 14 изображен снимок процессов через 8 минут после предыдущего. Можно увидеть, что время работы остановленных процессов не изменилось.

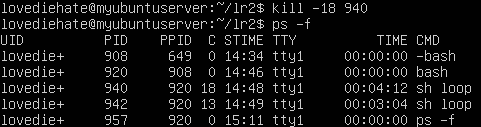


Рисунок 15 – Продолжение процесса

На Рисунке 15 заметно, что при продолжении процесса (команда kill –s CONT [PID] или kill -15 [PID]) работа возобновляется.

1. Часть II.
2. Запуск в консоли на выполнение трёх задач, двух в интерактивном режиме, одной - в фоновом

Использованные команды:

* jobs – просмотр текущих задач

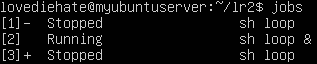


Рисунок 16 – Задачи

На Рисунке 16 показано, что запущено 3 задачи: 2 интерактивные и 1 фоновая. Интерактивные задачи пришлось приостановить.

1. Перевод интерактивной задачи в фоновую

Использованные команды:

* bg 3 – перевод задачи в фон



Рисунок 17 – Перевод задачи

На Рисунке 17 изображено использование команды bg для перевода задачи 3 в фоновый режим. На скриншоте видно, что задача 3 стала фоновой (появился амперсанд и задача запустилась).

1. Перевод из задачи из фонового режима в активный

Использованные команды:

* fg 2 – перевод задачи на передний план



Рисунок 18 – Перевод задачи

На Рисунке 18 с помощью команды fg задача 2 переведена в интерактивные. Для того, чтобы сделать снимок задач, пришлось приостановить действие процесса (^Z). Далее видно, что задача [2] переведена на передний план и приостановлена.

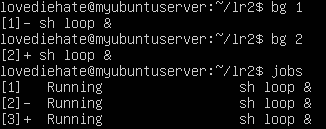


Рисунок 19 – Перевод всех задач на фон

1. 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.

Использованные команды:

* mkfifo fifo (создание канала)
* ls –l (просмотр файлов в текущем каталоге)

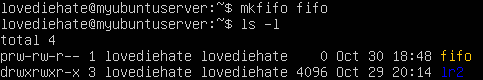


Рисунок 20 – Создание именованного канала

Буква p вначале говорит о том, что файл имеет тип именованный канал.

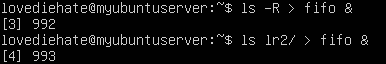


Рисунок 21 – Внесение списка и каталога в канал

Использованные команды:

* gzip -1 –c < fifo > my\_arch – настройка канала fifo

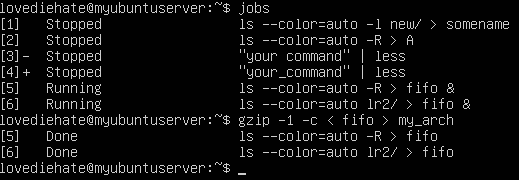


Рисунок 22 – Архивирование

На Рисунке 23 видно, что задачи 5 и 6 завершены успешно. Канал fifo был настроен на сжатие того, что туда попадёт и вывод в файл my\_arch. Задачи 5 и 6 передали данные в канал для архивации.

Использованные команды:

* zcat my\_arch – просмотр содержимого сжатого архива

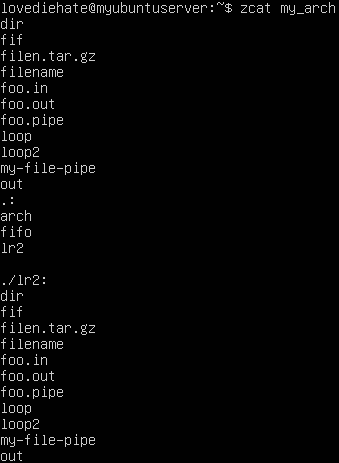


Рисунок 22 – Просмотр содержимого сжатого файла

Как можно увидеть на рисунке 25 – рекурсивный список файлов домашнего каталога и отдельный каталог успешно передались в fifo.

1. Часть III.

Вариант 2.

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

Использована команда:

* ps o user,uid,stat,c

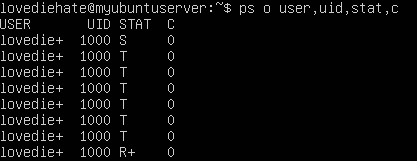


Рисунок 23 – Информация о процессах

На рисунке 23 выведена нужная информация по процессам: USER – имя владельца процесса, UID – ид владельца, STAT – статус процесса и C – приоритет процесса в планировщике.

1. 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.

Использована команда:

* pkill –2 –f (отправка процессу сигнала прерывания по имени)

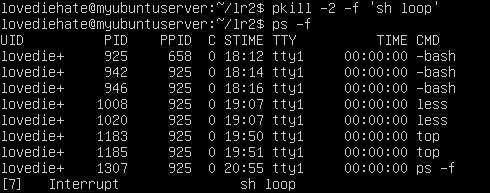


Рисунок 24 – Прерывание процесса

На Рисунке 24 заметно, что процесс sh loop завершил работу [Interrupt]. Параметр –f значит искать по полному названию файла, т.к. название с пробелом.

Используемые команды:

* kill -3 (отправка процессу сигнала 3 [QUIT])

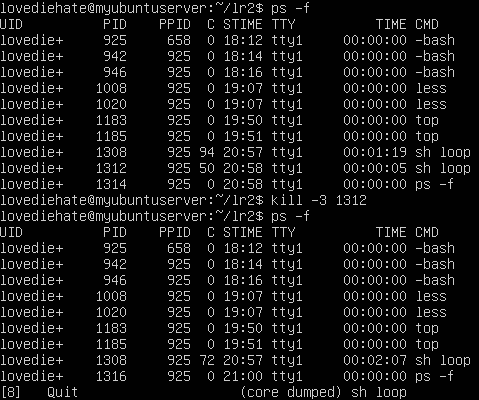


Рисунок 25 – Прерывание процесса 2

На последней строчке консоли, представленной на рисунке 25, видно, что процесс завершён с сообщением Quit, как мы и хотели.

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

Использована команда:

* id – выводит идентификатор текущего пользователя



Рисунок 26 – Информация о пользователе

На Рисунке 26 с помощью утилиты id узнаётся, какой идентификатор у текущего пользователя. В дальнейшем мы используем это значение для отрицания в поиске.

Использована команда:

* ps -afx --format=”comm pid uid uname gid group” | grep –v 1000 (вывод информации по процессу в заданном формате, исключая из результата строки, в которых попадается значение группы текущего пользователя)



Рисунок 27 – Строка ввода команды

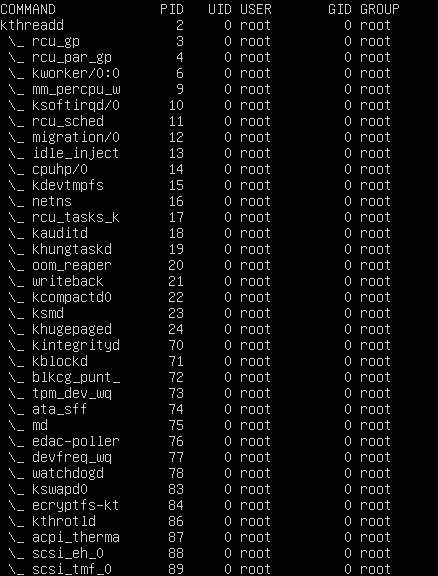


Рисунок 28 – Результат

На рисунке 27 выведены процессы, GID которых отличается от GID текущего пользователя.

# Вывод

Я ознакомился с понятием процесса в ОС Linux и приобрел базовые навыки работы с ним.