

Липецкий государственный технический университет

Кафедра прикладной математики

Отчет по лабораторной работе № 3
«Процессы в операционной системе Linux»
по курсу «Операционная система Linux»

Студент

подпись, дата

Богомолов Е.А.
фамилия, инициалы

Группа

Руководитель

Доцент, к. пед. наук
ученая степень, ученое звание

подпись, дата

Кургасов В.В.
фамилия, инициалы

Липецк 2021 г.

Содержание

Цель работы	3
Задание кафедры	4
1. Часть I	8
2. Часть II	17
3. Часть III	22
4. Часть IV	27
Выводы	31
Контрольные вопросы	32

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. Индивидуальные задания (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.

Часть IV:

1. Запустить программу виртуализации Oracle VM VirtualBox.
2. Запустить виртуальную машину Ubuntu.
3. Открыть окно интерпретатора команд.
4. Вывести общую информацию о системе.
 - (a) Вывести информацию о текущем интерпретаторе команд.
 - (b) Вывести информацию о текущем пользователе.
 - (c) Вывести информацию о текущем каталоге.
 - (d) Вывести информацию об оперативной памяти и области подкачки.
 - (e) Вывести информацию о дисковой памяти.
5. Выполнить команды получения информации о процессах
 - (a) Получить идентификатор текущего процесса(PID).
 - (b) Получить идентификатор родительского процесса(PPID).
 - (c) Получить идентификатор процесса инициализации системы.
 - (d) Получить информацию о выполняющихся процессах текущего пользователя в текущем интерпретаторе команд.
 - (e) Отобразить все процессы.
6. Выполнить команды управления процессами
 - (a) Получить информацию о выполняющихся процессах текущего пользователя в текущем интерпретаторе.
 - (b) Определить текущее значение `ps` по умолчанию.

- (c) Запустить интерпретатор `bash` с понижением приоритета `nice -n 10 bash`.
- (d) Определить PID запущенного интерпретатора.
- (e) Установить приоритет запущенного интерпретатора равным 5 `renice -n 5 <PID процесса>`.
- (f) Получить информацию о процессах `bash`.
`ps lax | grep bash`

Часть V:

1. Повторить команды `cat`, `head`, `tail`, `more`, `less`, `grep`, `find`
2. Разобраться с понятиями конвейер, перенаправление ввода-вывода.
3. Ознакомиться с информацией из рекомендованных источников(2) и других про конвейеризации.
4. Повторить назначение прав доступа. Команды `chmod`, `chown`.
5. Ознакомиться с информацией по теме процессы, посмотреть и опробовать примеры наиболее распространенных команд, изучить возможность запуска процессов в `supervisor`.
6. Изучить возможность автоматического запуска программ по расписанию.

1. Часть I

Задание:

1. Загрузиться не root, а пользователем.

```
ubuntu-server login: ubuntu-user
Password:
Welcome to Ubuntu 20.04.3 LTS (GNU/Linux 5.4.0-88-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

System information disabled due to load higher than 1.0

 * Super-optimized for small spaces - read how we shrank the memory
   footprint of MicroK8s to make it the smallest full K8s around.

   https://ubuntu.com/blog/microk8s-memory-optimisation

27 updates can be applied immediately.
To see these additional updates run: apt list --upgradable

Last login: Thu Nov  4 17:38:00 UTC 2021 on tty1
ubuntu-user@ubuntu-server:~$ _
```

Рисунок 1 – Загрузка не root, а пользователем.

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

```
ubuntu-user@ubuntu-server:~$ cd /boot
ubuntu-user@ubuntu-server:/boot$ ls
config-5.4.0-88-generic  initrd.img-5.4.0-88-generic  System.map-5.4.0-88-generic  vmlinuz.old
grub                    initrd.img.old               vmlinuz
initrd.img              lost+found                   vmlinuz-5.4.0-88-generic
ubuntu-user@ubuntu-server:/boot$ _
```

Рисунок 2 – Поиск файла с образом ядра.

Исходя из рис.2 делаем вывод, что номер версии Linux - 5.4.0.

3. Посмотреть процессы ps -f. Прокомментировать. Для этого почитать man ps.

```
ubuntu-user@ubuntu-server:/boot$ ps -f
UID          PID    PPID  C STIME TTY          TIME CMD
ubuntu-+    1156      659  0 21:03 tty1        00:00:00 -bash
ubuntu-+    1165    1156  0 21:04 tty1        00:00:00 ps -f
ubuntu-user@ubuntu-server:/boot$
```

Рисунок 3 – Просмотр процессов ps -f.

- UID – идентификатор пользователя.
 - PID – идентификатор процесса. Он принудительно назначается планировщиком при запуске процесса.
 - PPID – идентификатор родительского процесса.
 - C – численное значение расходования ресурсов процессора в процентах.
 - STIME – это время начала процесса.
 - TTY – имя управляющего терминала - терминала, с которого запущен процесс.
 - TIME – это общее время использования процессорного времени процессом.
 - CMD – команда, которой был запущен процесс, если программа не может прочитать аргументы процесса, он будет выведен в квадратных скобках.
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
```



```
ubuntu-user@ubuntu-server:/boot$ vi loop.sh
```

Рисунок 4 – Открытие редактора vi для файла loop.sh.

7. Посмотреть последовательно несколько раз `ps -f`. Записать сообщение, объяснить.

```
ubuntu-user@ubuntu-server:~$ ps -f
UID          PID    PPID  C STIME TTY          TIME CMD
ubuntu-+    1158      665  0 21:38 tty1      00:00:00 -bash
ubuntu-+    1194     1158  0 21:45 tty1      00:00:02 sh loop2.sh
ubuntu-+    1217     1158 53 21:56 tty1      00:00:04 sh loop2.sh
ubuntu-+    1218     1158  0 21:56 tty1      00:00:00 ps -f
ubuntu-user@ubuntu-server:~$ ps -f
UID          PID    PPID  C STIME TTY          TIME CMD
ubuntu-+    1158      665  0 21:38 tty1      00:00:00 -bash
ubuntu-+    1194     1158  0 21:45 tty1      00:00:02 sh loop2.sh
ubuntu-+    1217     1158 30 21:56 tty1      00:00:04 sh loop2.sh
ubuntu-+    1219     1158  0 21:56 tty1      00:00:00 ps -f
ubuntu-user@ubuntu-server:~$ ps -f
UID          PID    PPID  C STIME TTY          TIME CMD
ubuntu-+    1158      665  0 21:38 tty1      00:00:00 -bash
ubuntu-+    1194     1158  0 21:45 tty1      00:00:02 sh loop2.sh
ubuntu-+    1217     1158 25 21:56 tty1      00:00:04 sh loop2.sh
ubuntu-+    1220     1158  0 21:56 tty1      00:00:00 ps -f
ubuntu-user@ubuntu-server:~$ ps -f
UID          PID    PPID  C STIME TTY          TIME CMD
ubuntu-+    1158      665  0 21:38 tty1      00:00:00 -bash
ubuntu-+    1194     1158  0 21:45 tty1      00:00:02 sh loop2.sh
ubuntu-+    1217     1158 19 21:56 tty1      00:00:04 sh loop2.sh
ubuntu-+    1221     1158  0 21:56 tty1      00:00:00 ps -f
ubuntu-user@ubuntu-server:~$ _
```

Рисунок 10 – Просмотр `ps -f` последовательно.

Из рисунка 10 можно сделать вывод, что процент расхода ресурсов процессора уменьшается. Благодаря этому, можно сказать, что процессор тратит на этот процесс меньше ресурсов с течением времени.

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

```
ubuntu-user@ubuntu-server:~$ ps -f
UID          PID    PPID  C STIME TTY          TIME CMD
ubuntu-+    1346    1235  0 21:58 tty1        00:00:00 -bash
ubuntu-+    1361    1346 48 21:59 tty1        00:00:01 sh loop2.sh
ubuntu-+    1362    1346  0 21:59 tty1        00:00:00 ps -f
ubuntu-user@ubuntu-server:~$ kill -9 1361
[1]+  Killed                  sh loop2.sh
ubuntu-user@ubuntu-server:~$
```

Рисунок 11 – Уничтожение процесса loop2.sh при помощи сигнала kill -9 PID.

На рисунке 11 представлено удаление процесса и сообщение об успешном удалении.

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

```
ubuntu-user@ubuntu-server:~$ sh loop.sh&
[1] 1419
ubuntu-user@ubuntu-server:~$ ps -f
UID          PID    PPID  C STIME TTY          TIME CMD
ubuntu-+    1346    1235  0 21:58 tty1        00:00:00 -bash
ubuntu-+    1419    1346 99 22:16 tty1        00:00:45 sh loop.sh
ubuntu-+    1420    1346  0 22:17 tty1        00:00:00 ps -f
ubuntu-user@ubuntu-server:~$ ps -f
UID          PID    PPID  C STIME TTY          TIME CMD
ubuntu-+    1346    1235  0 21:58 tty1        00:00:00 -bash
ubuntu-+    1419    1346 99 22:16 tty1        00:00:56 sh loop.sh
ubuntu-+    1431    1346  0 22:17 tty1        00:00:00 ps -f
ubuntu-user@ubuntu-server:~$ ps -f
UID          PID    PPID  C STIME TTY          TIME CMD
ubuntu-+    1346    1235  0 21:58 tty1        00:00:00 -bash
ubuntu-+    1419    1346 99 22:16 tty1        00:01:01 sh loop.sh
ubuntu-+    1432    1346  0 22:17 tty1        00:00:00 ps -f
ubuntu-user@ubuntu-server:~$
```

Рисунок 12 – Запуск в фоне процесса loop: sh loop.sh&.

На основе рисунка 12 можно сделать вывод, что доля ресурсов, затраченных процессором, не уменьшается с течением времени. Из этого делаем заключение, что процесс работает.

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

```
ubuntu-+ 1346 1235 0 21:58 tty1 00:00:00 -bash
ubuntu-+ 1419 1346 99 22:16 tty1 00:01:01 sh loop.sh
ubuntu-+ 1432 1346 0 22:17 tty1 00:00:00 ps -f
ubuntu-user@ubuntu-server:~$
ubuntu-user@ubuntu-server:~$ kill -15 1419
ubuntu-user@ubuntu-server:~$ ps -f
UID      PID    PPID  C  STIME TTY          TIME CMD
ubuntu-+ 1346    1235  0  21:58 tty1      00:00:00 -bash
ubuntu-+ 1434    1346  0  22:18 tty1      00:00:00 ps -f
[1]+  Terminated                  sh loop.sh
ubuntu-user@ubuntu-server:~$
```

Рисунок 13 – Завершение процесса loop командой kill -15 PID.

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

```
ubuntu-user@ubuntu-server:~$ sh loop.sh&
[1] 1436
ubuntu-user@ubuntu-server:~$ kill -9 1436
ubuntu-user@ubuntu-server:~$ ps -f
UID      PID    PPID  C  STIME TTY          TIME CMD
ubuntu-+ 1346    1235  0  21:58 tty1      00:00:00 -bash
ubuntu-+ 1438    1346  0  22:27 tty1      00:00:00 ps -f
[1]+  Killed                        sh loop.sh
ubuntu-user@ubuntu-server:~$
```

Рисунок 14 – Запуск процесса и его уничтожение командой kill -9 PID.

12. Запустить еще один экземпляр оболочки: bash.

```
ubuntu-user@ubuntu-server:~$ bash
ubuntu-user@ubuntu-server:~$ ps -f
UID      PID    PPID  C  STIME TTY          TIME CMD
ubuntu-+ 1346    1235  0  21:58 tty1      00:00:00 -bash
ubuntu-+ 1440    1346  0  22:32 tty1      00:00:00 bash
ubuntu-+ 1446    1440  0  22:32 tty1      00:00:00 ps -f
ubuntu-user@ubuntu-server:~$ _
```

Рисунок 15 – Запуск экземпляра оболочки bash.

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

```
ubuntu-user@ubuntu-server:~$ sh loop.sh&
[1] 1200
ubuntu-user@ubuntu-server:~$ sh loop.sh&
[2] 1201
ubuntu-user@ubuntu-server:~$ sh loop.sh&
[3] 1202
ubuntu-user@ubuntu-server:~$ ps -f
UID          PID    PPID  C STIME TTY          TIME CMD
ubuntu-+    1153      665  0 22:42 tty1        00:00:00 -bash
ubuntu-+    1200    1153 65 22:50 tty1        00:00:11 sh loop.sh
ubuntu-+    1201    1153 45 22:50 tty1        00:00:04 sh loop.sh
ubuntu-+    1202    1153 33 22:50 tty1        00:00:01 sh loop.sh
ubuntu-+    1203    1153  0 22:50 tty1        00:00:00 ps -f
ubuntu-user@ubuntu-server:~$ kill -9 1200
ubuntu-user@ubuntu-server:~$ kill -9 1201
[1] Killed                  sh loop.sh
ubuntu-user@ubuntu-server:~$ kill -9 1202
[2]- Killed                  sh loop.sh
ubuntu-user@ubuntu-server:~$ ps -f
UID          PID    PPID  C STIME TTY          TIME CMD
ubuntu-+    1153      665  0 22:42 tty1        00:00:00 -bash
ubuntu-+    1204    1153  0 22:51 tty1        00:00:00 ps -f
[3]+ Killed                  sh loop.sh
ubuntu-user@ubuntu-server:~$
```

Рисунок 16 – Запуск нескольких процессов в фоне.

2. Часть II

Задание:

1. Запустить в консоли на выполнение три задачи, две в интерактивном режиме, одну - в фоновом.

Запустим задачи с помощью команд (рис. 17):

- sh loop.sh (интерактивный режим)
- sh loop.sh (интерактивный режим)
- sh loop.sh (фоновый режим)

```
ubuntu-user@ubuntu-server:~$ sh loop.sh
^Z
[1]+  Stopped                  sh loop.sh
ubuntu-user@ubuntu-server:~$ sh loop.sh
^Z
[2]+  Stopped                  sh loop.sh
ubuntu-user@ubuntu-server:~$ sh loop.sh&
[3] 1890
ubuntu-user@ubuntu-server:~$ _
```

Рисунок 17 – Запуск трех задач.

С помощью команды jobs получим список процессов в текущей оболочке (рис. 18):

- jobs (список процессов в текущей оболочке)

```
ubuntu-user@ubuntu-server:~$ jobs
[1]-  Stopped                  sh loop.sh
[2]+  Stopped                  sh loop.sh
[3]   Running                  sh loop.sh &
ubuntu-user@ubuntu-server:~$ _
```

Рисунок 18 – Использование команды jobs.

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

С помощью команды bg и номера, присвоенного задаче командной оболочкой при остановке ее исполнения, переведем задачу из интерактивного режима в фоновый (рис. 19):

- bg %1 (возобновление задачи №1)

```

ubuntu-user@ubuntu-server:~$ sh loop.sh
^Z
[1]+  Stopped                  sh loop.sh
ubuntu-user@ubuntu-server:~$ sh loop.sh
^Z
[2]+  Stopped                  sh loop.sh
ubuntu-user@ubuntu-server:~$ sh loop.sh&
[3] 1890
ubuntu-user@ubuntu-server:~$ jobs
[1]-  Stopped                  sh loop.sh
[2]+  Stopped                  sh loop.sh
[3]   Running                  sh loop.sh &
ubuntu-user@ubuntu-server:~$ bg %1
[1]-  sh loop.sh &
ubuntu-user@ubuntu-server:~$ jobs
[1]   Running                  sh loop.sh &
[2]+  Stopped                  sh loop.sh
[3]-  Running                  sh loop.sh &
ubuntu-user@ubuntu-server:~$ _

```

Рисунок 19 – Перевод задачи в фоновый режим.

- jobs (список процессов в текущей оболочке)

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

С помощью команды `fg` и номера, присвоенного задаче командной оболочкой при остановке ее исполнения, переведем задачу из фонового режима в интерактивный, а также с помощью команды `bg` делаем полностью обратное действие (рис. 19):

- `fg %1` (возобновление задачи №1 в интерактивном режиме)
- `bg %1` (возобновление задачи №1 в фоновом режиме)
- `bg %2` (возобновление задачи №1 в фоновом режиме)
- jobs (список процессов в текущей оболочке)

```

ubuntu-user@ubuntu-server:~$ jobs
[1]  Running                  sh loop.sh &
[2]+ Stopped                  sh loop.sh
[3]-  Running                  sh loop.sh &
ubuntu-user@ubuntu-server:~$ fg %1
sh loop.sh
^Z
[1]+  Stopped                  sh loop.sh
ubuntu-user@ubuntu-server:~$ bg %1
[1]+ sh loop.sh &
ubuntu-user@ubuntu-server:~$ bg %2
[2]+ sh loop.sh &
ubuntu-user@ubuntu-server:~$ jobs
[1]  Running                  sh loop.sh &
[2]-  Running                  sh loop.sh &
[3]+  Running                  sh loop.sh &
ubuntu-user@ubuntu-server:~$

```

Рисунок 20 – Перевод задачи в фоновый режим.

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.

Создадим именованный канал для архивирования с помощью команды `mkfifo`. Посмотрим, что получилось в результате работы команды с помощью `ls -l`. Результат выполнения представлен на рис. 21.

- `mkfifo myChannel` (создание именованного канала с именем «myChannel»)
- `ls -l myChannel` (проверка создания файла)
- `gzip -9 -c < myChannel > out.gz` (передача дом. каталога)
- `zcat out.gz` (просмотр сжатых файлов)
- `tar -cvf out.tar /home > myChannel` (передача каталога с подкаталогами и файлами)
- `zcat out.gz` (просмотр сжатых файлов)

```

ubuntu-user@ubuntu-server:~$ mkfifo myChannel
ubuntu-user@ubuntu-server:~$ ls -l myChannel
prw-rw-r-- 1 ubuntu-user ubuntu-user 0 Nov  9 05:10 myChannel
ubuntu-user@ubuntu-server:~$ _

```

Рисунок 21 – создание именованного канала и проверка.

```

root@ubuntu-server:/home/ubuntu-user# gzip -9 -c < myChannel > out.gz &
[2] 1463
root@ubuntu-server:/home/ubuntu-user# ls -R > myChannel
[2]- Done      gzip -9 -c < myChannel > out.gz
root@ubuntu-server:/home/ubuntu-user# zcat out.gz
.:
loop2.sh
loop.sh
myChannel
MYDIR
out.gz

./MYDIR:
MIFILE1
MYDIR1
MYDIR2
MYDIR3
MYFILE3

./MYDIR/MYDIR1:

./MYDIR/MYDIR2:
MYFILE2

./MYDIR/MYDIR3:
root@ubuntu-server:/home/ubuntu-user# _

```

Рисунок 22 – Передача списка файлов домашнего каталога пользователя «ubuntu-user» и проверка.

```

root@ubuntu-server:/home/ubuntu-user# mkdir MYDIR
root@ubuntu-server:/home/ubuntu-user# cd MYDIR
root@ubuntu-server:/home/ubuntu-user/MYDIR# mkdir MYDIR1
root@ubuntu-server:/home/ubuntu-user/MYDIR# mkdir MYDIR2
root@ubuntu-server:/home/ubuntu-user/MYDIR# mkdir MYDIR3
root@ubuntu-server:/home/ubuntu-user/MYDIR# touch MIFILE1
root@ubuntu-server:/home/ubuntu-user/MYDIR# cd MYDIR2
root@ubuntu-server:/home/ubuntu-user/MYDIR/MYDIR2# touch MYFILE2
root@ubuntu-server:/home/ubuntu-user/MYDIR/MYDIR2# cd ..
root@ubuntu-server:/home/ubuntu-user/MYDIR# touch MYFILE3
root@ubuntu-server:/home/ubuntu-user/MYDIR# ls -l
total 12
-rw-r--r-- 1 root root    0 Nov 12 13:53 MIFILE1
drwxr-xr-x 2 root root 4096 Nov 12 13:52 MYDIR1
drwxr-xr-x 2 root root 4096 Nov 12 13:53 MYDIR2
drwxr-xr-x 2 root root 4096 Nov 12 13:52 MYDIR3
-rw-r--r-- 1 root root    0 Nov 12 13:53 MYFILE3
root@ubuntu-server:/home/ubuntu-user/MYDIR# cd ..
root@ubuntu-server:/home/ubuntu-user# _

```

Рисунок 23 – Создание нового каталога с файлами и подкаталогами.

```

root@ubuntu-server:/home/ubuntu-user# gzip -9 -c < myChannel > out.gz &
[2] 1468
root@ubuntu-server:/home/ubuntu-user# tar -cvf out.tar MYDIR > myChannel
[2]- Done      gzip -9 -c < myChannel > out.gz

```

Рисунок 24 – Передача всего каталога с подкаталогами.

```
root@ubuntu-server:/home/ubuntu-user# zcat out.gz
MYDIR/
MYDIR/MYDIR2/
MYDIR/MYDIR2/MYFILE2
MYDIR/MYDIR3/
MYDIR/MYFILE3
MYDIR/MIFILE1
MYDIR/MYDIR1/
root@ubuntu-server:/home/ubuntu-user# _
```

Рисунок 25 – Просмотр результатов.

3. Часть III

Вариант 2:

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

Команда `top` дает представление о динамике процессов. Она выводит список процессов, отсортированный по количеству занятой памяти или использованного процессорного времени, и обновляет его через указанные промежутки времени (по умолчанию через каждые 3 секунды). С помощью параметра `-u` выведем только те процессы, которые запущены от имени введенного пользователя.

- `top -u ubuntu-user` (рис. 22)

```
top - 22:04:39 up 7 min, 1 user, load average: 0.02, 0.52, 0.38
Tasks: 97 total, 1 running, 96 sleeping, 0 stopped, 0 zombie
%Cpu(s): 0.0 us, 0.3 sy, 0.0 ni, 99.7 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st
MiB Mem : 981.2 total, 161.1 free, 183.3 used, 636.8 buff/cache
MiB Swap: 1735.0 total, 1735.0 free, 0.0 used, 645.9 avail Mem
```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
1446	ubuntu-+	20	0	18512	9808	8200	S	0.0	1.0	0:00.17	systemd
1447	ubuntu-+	20	0	103280	3336	16	S	0.0	0.3	0:00.00	(sd-pam)
1452	ubuntu-+	20	0	8264	5280	3572	S	0.0	0.5	0:00.05	bash

Рисунок 26 – Использование команды `top`.

С помощью команды `f` перейдем к управлению полями.

```
Fields Management for window 1:Def, whose current sort field is %CPU
  Navigate with Up/Dn, Right selects for move then <Enter> or Left commits,
  'd' or <Space> toggles display, 's' sets sort. Use 'q' or <Esc> to end!

* PID      = Process Id
* USER     = Effective User Name
* PR       = Priority
* NI       = Nice Value
* VIRT     = Virtual Image (KiB)
* RES      = Resident Size (KiB)
* SHR      = Shared Memory (KiB)
* S        = Process Status
* %CPU     = CPU Usage
* %MEM     = Memory Usage (RES)
* TIME+    = CPU Time, hundredths
* COMMAND  = Command Name/Line
  PPID     = Parent Process pid
  UID      = Effective User Id
  RUID     = Real User Id
  RUSER    = Real User Name
  SUID     = Saved User Id
  SUSER    = Saved User Name
  GID      = Group Id
  GROUP    = Group Name
  PGRP     = Process Group Id
  TTY      = Controlling Tty
  TPGID    = Tty Process Grp Id
  SID      = Session Id
  nTH      = Number of Threads
  P        = Last Used Cpu (SMP)
  TIME     = CPU Time
  SWAP     = Swapped Size (KiB)
  CODE     = Code Size (KiB)
  DATA    = Data+Stack (KiB)
  nMaj     = Major Page Faults
  nMin     = Minor Page Faults
  nDRT     = Dirty Pages Count
  WCHAN    = Sleeping in Function
  Flags    = Task Flags <sched.h>
  CGROUPS  = Control Groups
  SUPGIDS  = Supp Groups IDs
  SUPGRPS  = Supp Groups Names
  TGID     = Thread Group Id
  OOMa     = OOMEM Adjustment
  OOMs     = OOMEM Score current
  ENVIRON  = Environment vars
  vMj      = Major Faults delta
  vMn      = Minor Faults delta
  USED     = Res+Swap Size (KiB)
  nsIPC    = IPC namespace Inode
  nsMNT    = MNT namespace Inode
  nsNET    = NET namespace Inode
  nsPID    = PID namespace Inode
  nsUSER   = USER namespace Inode
  nsUTS    = UTS namespace Inode
  LXC      = LXC container name
  RSan     = RES Anonymous (KiB)
  RSfd     = RES File-based (KiB)
  RSlk     = RES Locked (KiB)
  RSsh     = RES Shared (KiB)
  CGNAME   = Control Group name
  NU       = Last Used NUMA node
```

Рисунок 27 – Управление доступными полями.

С помощью команды `d` отберем только необходимые поля и удалим ненужные.

```
Fields Management for window 1:Def, whose current sort field is %CPU
  Navigate with Up/Dn, Right selects for move then <Enter> or Left commits,
  'd' or <Space> toggles display, 's' sets sort. Use 'q' or <Esc> to end!

* PID = Process Id
* USER = Effective User Name
* PR = Priority
  NI = Nice Value
  VIRT = Virtual Image (KiB)
  RES = Resident Size (KiB)
  SHR = Shared Memory (KiB)
* S = Process Status
  %CPU = CPU Usage
  %MEM = Memory Usage (RES)
  TIME+ = CPU Time, hundredths
  COMMAND = Command Name/Line
  PPID = Parent Process pid
  UID = Effective User Id
  RUID = Real User Id
  RUSER = Real User Name
  SUID = Saved User Id
  SUSER = Saved User Name
  GID = Group Id
  GROUP = Group Name
  PGRP = Process Group Id
  TTY = Controlling Tty
  TPGID = Tty Process Grp Id
  SID = Session Id
  nTH = Number of Threads
  P = Last Used Cpu (SMP)
  TIME = CPU Time
  SWAP = Swapped Size (KiB)
  CODE = Code Size (KiB)
  DATA = Data+Stack (KiB)
  nMaj = Major Page Faults
  nMin = Minor Page Faults
  nDRT = Dirty Pages Count

  WCHAN = Sleeping in Function
  Flags = Task Flags <sched.h>
  CGROUPS = Control Groups
  SUPGIDS = Supp Groups IDs
  SUPGRPS = Supp Groups Names
  TGID = Thread Group Id
  OOMa = OOMEM Adjustment
  OOMs = OOMEM Score current
  ENVIRON = Environment vars
  vMj = Major Faults delta
  vMn = Minor Faults delta
  USED = Res+Swap Size (KiB)
  nsIPC = IPC namespace Inode
  nsMNT = MNT namespace Inode
  nsNET = NET namespace Inode
  nsPID = PID namespace Inode
  nsUSER = USER namespace Inode
  nsUTS = UTS namespace Inode
  LXC = LXC container name
  RSan = RES Anonymous (KiB)
  RSfd = RES File-based (KiB)
  RSlk = RES Locked (KiB)
  RSsh = RES Shared (KiB)
  CGNAME = Control Group name
  NU = Last Used NUMA node
```

Рисунок 28 – создание именованного канала и проверка.

Результат работы всех действий представлен на рис. 25.

```
top - 22:22:50 up 26 min,  1 user,  load average: 0.01, 0.03, 0.10
Tasks: 101 total,   1 running,  96 sleeping,   4 stopped,   0 zombie
%Cpu(s):  0.0 us,   0.0 sy,   0.0 ni, 99.7 id,   0.0 wa,   0.0 hi,   0.3 si,   0.0 st
MiB Mem :   981.2 total,   159.2 free,   184.3 used,   637.7 buff/cache
MiB Swap:  1735.0 total,  1735.0 free,    0.0 used.   644.7 avail Mem

  PID USER      PR S
 1503 ubuntu-+  20 R
 1446 ubuntu-+  20 S
 1447 ubuntu-+  20 S
 1452 ubuntu-+  20 S
 1467 ubuntu-+  20 T
 1476 ubuntu-+  20 T
 1477 ubuntu-+  20 T
 1479 ubuntu-+  20 T
```

Рисунок 29 – создание именованного канала и проверка.

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.

- sh loop.sh&
- sh loop.sh&
- ps -f
- kill -2 1177 (SIGINT)
- kill -3 1178 (SIGQUIT)
- ps -f

```

ubuntu-user@ubuntu-server:~$ sh loop.sh&
[1] 1177
ubuntu-user@ubuntu-server:~$ sh loop.sh&
[2] 1178
ubuntu-user@ubuntu-server:~$ ps -f
UID          PID    PPID  C STIME TTY          TIME CMD
ubuntu-+    1167      656  0 14:26 tty1        00:00:00 -bash
ubuntu-+    1177     1167  84 14:26 tty1        00:00:08 sh loop.sh
ubuntu-+    1178     1167  51 14:26 tty1        00:00:02 sh loop.sh
ubuntu-+    1179     1167  0 14:26 tty1        00:00:00 ps -f
ubuntu-user@ubuntu-server:~$ kill -2 1177
ubuntu-user@ubuntu-server:~$ kill -3 1178
[1]-  Interrupt                sh loop.sh
ubuntu-user@ubuntu-server:~$ ps -f
UID          PID    PPID  C STIME TTY          TIME CMD
ubuntu-+    1167      656  0 14:26 tty1        00:00:00 -bash
ubuntu-+    1182     1167  0 14:27 tty1        00:00:00 ps -f
[2]+  Quit                      (core dumped) sh loop.sh
ubuntu-user@ubuntu-server:~$ _

```

Рисунок 30 – Выполнения задания.

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

- `ps -ao pid,cmd,gid|grep -v 1000`

С помощью опции команды `ps` – `-a` выведем все процессы, а с помощью `-o` – отфильтруем вывод. С помощью команды `grep -v` отфильтруем строки так, чтобы в выводе были все строки за исключением, содержащих данный образец.

```

for more details see ps(1).
ubuntu-user@ubuntu-server:~$ ps -ao pid,cmd,gid
  PID CMD          GID
 1225 -bash          1000
 1241 ps -ao pid,cmd,gid 1000
ubuntu-user@ubuntu-server:~$ ps -ao pid,cmd,gid|grep -v 1000
  PID CMD          GID
ubuntu-user@ubuntu-server:~$ _

```

Рисунок 31 – Выполнения задания.

4. Часть IV

1. Запустить программу виртуализации Oracle VM VirtualBox.
2. Запустить виртуальную машину Ubuntu.
3. Открыть окно интерпретатора команд.
4. Вывести общую информацию о системе.

- (a) Вывести информацию о текущем интерпретаторе команд.

```
echo $SHELL
```

- (b) Вывести информацию о текущем пользователе.

```
whoami
```

- (c) Вывести информацию о текущем каталоге.

```
pwd
```

- (d) Вывести информацию об оперативной памяти и области подкачки.

```
free
```

- (e) Вывести информацию о дисковой памяти.

```
df
```

```

ubuntu-user@ubuntu-server:~$ echo $SHELL
/bin/bash
ubuntu-user@ubuntu-server:~$ whoami
ubuntu-user
ubuntu-user@ubuntu-server:~$ pwd
/home/ubuntu-user
ubuntu-user@ubuntu-server:~$ free
               total        used        free      shared  buff/cache   available
Mem:           1004796       238396         71208          1016        695192        596692
Swap:          1776636           3852       1772784
ubuntu-user@ubuntu-server:~$ df
Filesystem            1K-blocks    Used Available Use% Mounted on
udev                  457236      0      457236   0% /dev
tmpfs                 100480     1124      99356   2% /run
/dev/mapper/ubantu--vg-ubantu--lv 9219412 5508040   3223336   64% /
tmpfs                 502396      0      502396   0% /dev/shm
tmpfs                  5120       0         5120   0% /run/lock
tmpfs                 502396      0      502396   0% /sys/fs/cgroup
/dev/loop0            56832     56832      0 100% /snap/core18/2246
/dev/loop1            56832     56832      0 100% /snap/core18/2128
/dev/loop3            119424    119424      0 100% /snap/docker/1125
/dev/loop6            68864     68864      0 100% /snap/lxd/21835
/dev/loop5            33280     33280      0 100% /snap/snapd/13640
/dev/loop4            63360     63360      0 100% /snap/core20/1169
/dev/loop7            43264     43264      0 100% /snap/snapd/13831
/dev/loop2            68864     68864      0 100% /snap/lxd/21803
/dev/sda2             999320    124976    805532   14% /boot
/dev/loop8            63360     63360      0 100% /snap/core20/1242
tmpfs                 100476      0      100476   0% /run/user/1000
ubuntu-user@ubuntu-server:~$ _

```

Рисунок 32 – Общая информация о системе.

5. Выполнить команды получения информации о процессах

- (a) Получить идентификатор текущего процесса(PID).

```
echo $$
```

- (b) Получить идентификатор родительского процесса(PPID).

```
echo $PPID
```

- (c) Получить идентификатор процесса инициализации системы.

```
pidof init
```

- (d) Получить информацию о выполняющихся процессах текущего пользователя в текущем интерпретаторе команд.

```
ps
```

- (e) Отобразить все процессы.

```
ps -e
```

```

ubuntu-user@ubuntu-server:~$ echo $$
1316
ubuntu-user@ubuntu-server:~$ echo $PPID
658
ubuntu-user@ubuntu-server:~$ pidof init
1
ubuntu-user@ubuntu-server:~$ ps
  PID TTY          TIME CMD
  1316 tty1      00:00:00 bash
  14388 tty1      00:00:00 ps
ubuntu-user@ubuntu-server:~$ ps -e

```

Рисунок 33 – Получение информации о процессах.

```

551 ?          00:00:00 jbd2/sda2-8
552 ?          00:00:00 ext4-rsv-conver
565 ?          00:00:00 systemd-timesyn
613 ?          00:00:00 systemd-network
615 ?          00:00:00 systemd-resolve
627 ?          00:00:00 accounts-daemon
630 ?          00:00:00 cron
631 ?          00:00:00 dbus-daemon
639 ?          00:00:00 networkd-dispat
640 ?          00:00:00 rsyslogd
641 ?          00:00:06 dockerd
643 ?          00:00:07 snapd
644 ?          00:00:00 systemd-logind
646 ?          00:00:00 udisksd
649 ?          00:00:00 atd
658 tty1       00:00:00 login
677 ?          00:00:29 kworker/0:6-events
694 ?          00:00:00 unattended-upgr
702 ?          00:00:00 polkitd
892 ?          00:00:07 containerd
980 ?          00:00:00 none
1099 ?         00:00:00 loop8
1310 ?         00:00:00 systemd
1311 ?         00:00:00 (sd-pam)
1316 tty1      00:00:00 bash
1332 ?         00:00:02 kworker/u2:1-events_power_efficient
1533 ?         00:00:02 kworker/u2:3-events_unbound
11986 ?        00:00:00 xfsalloc
11993 ?        00:00:00 xfs_mru_cache
11999 ?        00:00:00 jfsIO
12000 ?        00:00:00 jfsCommit
12001 ?        00:00:00 jfsSync
14363 ?        00:00:00 kworker/u2:0-events_power_efficient
14369 ?        00:00:00 kworker/0:1-events
14385 ?        00:00:00 kworker/0:0
14389 tty1     00:00:00 ps
ubuntu-user@ubuntu-server:~$

```

Рисунок 34 – Все процессы.

6. Выполнить команды управления процессами

- (a) Получить информацию о выполняющихся процессах текущего пользователя в текущем интерпретаторе.

```
ps
```

- (b) Определить текущее значение nice по умолчанию.

```
nice
```

- (c) Запустить интерпретатор bash с понижением приоритета nice -n 10 bash.

```
nice -n 10 bash
```

- (d) Определить PID запущенного интерпретатора.

```
echo $$
```

- (e) Установить приоритет запущенного интерпретатора равным 5 renice -n 5 -p <PID процесса>.

- (f) Получить информацию о процессах bash.

```
ps lax | grep bash
```

```
ubuntu-user@ubuntu-server:~$ ps
  PID TTY          TIME CMD
  1316 tty1      00:00:00 bash
 14395 tty1      00:00:00 bash
 14408 tty1      00:00:00 ps
ubuntu-user@ubuntu-server:~$ nice
10
ubuntu-user@ubuntu-server:~$ nice -n 10 bash
ubuntu-user@ubuntu-server:~$ echo $$
14410
ubuntu-user@ubuntu-server:~$ renice -n 5 -p 14410
renice: failed to set priority for 14410 (process ID): Permission denied
ubuntu-user@ubuntu-server:~$ renice -n -5 14410
renice: failed to set priority for 14410 (process ID): Permission denied
ubuntu-user@ubuntu-server:~$ ps lax| grep bash
 4 1000    1316      658 20   0  8264  4956 do_wai S   tty1      0:00 -bash
 0 1000    14395     1316 30  10  8272  5172 do_wai SN  tty1      0:00  bash
 0 1000    14410     14395 39  19  8272  5024 do_wai SN  tty1      0:00  bash
 0 1000    14419     14410 39  19  6300   740 -      RN+  tty1      0:00 grep --color=auto  bash
ubuntu-user@ubuntu-server:~$
```

Рисунок 35 – Все процессы.

Выводы

В ходе выполнения данной лабораторной работы мной были получены знания о понятии процесса, приобретен опыт и навыки управления процессами в операционной системе Linux.

Контрольные вопросы

1. Перечислите состояния задачи в ОС Ubuntu.

- running (выполнение) – после выделения ей процессора.
- sleeping (спячка) – при блокировке экрана.
- stopped (остановлена) – выполнение задачи прекращено, но из системы не удалена.
- dead (смерть) – может быть удалена из системы.
- active (активный) – используются при планировании выполнения процесса.
- expired (неактивный) – используются при планировании выполнения процесса.

2. Как создаются задачи в ОС Ubuntu?

Задачи создаются путем вызова функции clone.

3. Назовите классы потоков в ОС Ubuntu.

- Потоки реального времени, обслуживаемые по алгоритму FIFO.
- Потоки реального времени, обслуживаемые в порядке циклической очереди.
- Потоки разделения времени.

4. Как используется приоритет планирования при запуске задачи.

У каждого потока есть приоритет планирования. Значение по умолчанию равно 20, но оно может быть изменено при помощи системного вызова nice(value), вычитающего значение value из 20. Поскольку value должно находиться в диапазоне от -20 до +19, приоритеты всегда попадают в промежуток от 1 до 40.

5. Как можно изменить приоритет планирования для выполняющейся задачи?

с помощью команды nice.