

Липецкий государственный технический университет
Факультет автоматизации и информатики
Кафедра автоматизированных систем управления

ЛАБОРАТОРНАЯ РАБОТА №2

по курсу «ОС Linux»

Понятие о процессе в ОС Linux

Студент

Печенкин Д.В.

Группа ПИ-18

Руководитель

Кургасов В.В.

Доцент

Липецк 2020

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),
 - о 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

- 1) Отобразить информацию о процессах указанного пользователя в виде иерархии, вывод отсортировать по значениям PID.
- 2) С помощью сигнала SIGSTOP приостановить выполнение процесса, владельцем которого является текущий пользователь. Через несколько секунд возобновить выполнение процесса.
- 3) Определить идентификаторы и имена процессов, не связанных с указанным терминалом.
- 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

Найти файл с образом ядра. Выяснить по имени файла номер версии Linux (от имени пользователя).

```
danibrogue@jorgserver:/$ uname -a
Linux jorgserver 5.4.0-48-generic #52-Ubuntu SMP Thu Sep 10 10:58:49 UTC 2020 x86_64 x86_64 x86_64 GNU/Linux
danibrogue@jorgserver:/$ _
```

Рисунок 1 – версия Linux

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

```
danibrogue@jorgserver:/$ ps -f
UID      PID  PPID  C STIME TTY          TIME CMD
danibrogue+ 1262    870  0 07:37 pts/1    00:00:00 -bash
danibrogue+ 1361    1262  0 07:56 pts/1    00:00:00 ps -f
danibrogue@jorgserver:/$ _
```

Рисунок 2 – процессы

От имени пользователя запущены 2 процесса: оболочка bash и вывод процессов. Для процесса вывода родительским является оболочка. Процессы запущены с одного терминала с разницей в 19 секунд.

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

Для активации текстового редактора была использована команда “vi <имя файла>”

```
while true; do true; done
```

Рисунок 3 – создание сценария loop

```
while true; do true; echo 'Hello'; done
```

Рисунок 4 – создание сценария loop2

Запустить loop2 на переднем плане: sh loop2. Остановить, послав сигнал STOP. Посмотреть последовательно несколько раз ps –f. Записать сообщение, объяснить.

```
Hello
Hello
Hello
Hello
^Z
[1]+  Stopped                  sh loop2.txt
danibrogue@jorgserver:~$ ps -f
UID      PID  PPID  C STIME TTY          TIME CMD
danibrogue+ 913    638  0 11:43 tty1    00:00:00 -bash
danibrogue+ 1011   913  74 12:00 tty1    00:00:09 sh loop2.txt
danibrogue+ 1012   913  0 12:00 tty1    00:00:00 ps -f
danibrogue@jorgserver:~$ ps -f
UID      PID  PPID  C STIME TTY          TIME CMD
danibrogue+ 913    638  0 11:43 tty1    00:00:00 -bash
danibrogue+ 1011   913  40 12:00 tty1    00:00:09 sh loop2.txt
danibrogue+ 1013   913  0 12:00 tty1    00:00:00 ps -f
danibrogue@jorgserver:~$ ps -f
UID      PID  PPID  C STIME TTY          TIME CMD
danibrogue+ 913    638  0 11:43 tty1    00:00:00 -bash
danibrogue+ 1011   913  27 12:00 tty1    00:00:09 sh loop2.txt
danibrogue+ 1014   913  0 12:00 tty1    00:00:00 ps -f
danibrogue@jorgserver:~$ ps -f
UID      PID  PPID  C STIME TTY          TIME CMD
danibrogue+ 913    638  0 11:43 tty1    00:00:00 -bash
danibrogue+ 1011   913  21 12:00 tty1    00:00:09 sh loop2.txt
danibrogue+ 1015   913  0 12:01 tty1    00:00:00 ps -f
danibrogue@jorgserver:~$ ps -f
UID      PID  PPID  C STIME TTY          TIME CMD
danibrogue+ 913    638  0 11:43 tty1    00:00:00 -bash
danibrogue+ 1011   913  17 12:00 tty1    00:00:09 sh loop2.txt
danibrogue+ 1016   913  0 12:01 tty1    00:00:00 ps -f
danibrogue@jorgserver:~$ ps -f
UID      PID  PPID  C STIME TTY          TIME CMD
danibrogue+ 913    638  0 11:43 tty1    00:00:00 -bash
danibrogue+ 1011   913  14 12:00 tty1    00:00:09 sh loop2.txt
danibrogue+ 1017   913  0 12:01 tty1    00:00:00 ps -f
danibrogue@jorgserver:~$
```

Рисунок 5 – работа с процессом loop2

Вывод информации о процессах происходил примерно каждые 10 секунд. Мы можем видеть, что за время действия процесса он занял некоторое количество ресурсов компьютера, но по мере течения времени после остановки он постепенно их высвобождает.

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

Прокомментировать.

```
danibrogue@jorgserver:~$ kill -9 1011
[1]+  Killed                  sh loop2.txt
danibrogue@jorgserver:~$ _
```

Рисунок 6 – убийство процесса loop2

После того, как мы дали команду убить процесс loop2, мы получили сообщение “Killed” и название процесса.

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

```
danibrogue@jorgserver:~$ sh loop.txt&
[1] 1028
danibrogue@jorgserver:~$ ps -f
UID      PID  PPID  C STIME TTY          TIME CMD
danibro+    913    638  0 11:43 tty1        00:00:00 -bash
danibro+   1028    913  86 12:08 tty1      00:00:04 sh loop.txt
danibro+   1029    913  0 12:08 tty1      00:00:00 ps -f
danibrogue@jorgserver:~$ ps -f
UID      PID  PPID  C STIME TTY          TIME CMD
danibro+    913    638  0 11:43 tty1        00:00:00 -bash
danibro+   1028    913  96 12:08 tty1      00:00:15 sh loop.txt
danibro+   1030    913  0 12:08 tty1      00:00:00 ps -f
danibrogue@jorgserver:~$ ps -f
UID      PID  PPID  C STIME TTY          TIME CMD
danibro+    913    638  0 11:43 tty1        00:00:00 -bash
danibro+   1028    913  96 12:08 tty1      00:00:25 sh loop.txt
danibro+   1031    913  0 12:08 tty1      00:00:00 ps -f
danibrogue@jorgserver:~$ ps -f
UID      PID  PPID  C STIME TTY          TIME CMD
danibro+    913    638  0 11:43 tty1        00:00:00 -bash
danibro+   1028    913  97 12:08 tty1      00:00:36 sh loop.txt
danibro+   1032    913  0 12:08 tty1      00:00:00 ps -f
danibrogue@jorgserver:~$ ps -f
UID      PID  PPID  C STIME TTY          TIME CMD
danibro+    913    638  0 11:43 tty1        00:00:00 -bash
danibro+   1028    913  97 12:08 tty1      00:00:45 sh loop.txt
danibro+   1033    913  0 12:08 tty1      00:00:00 ps -f
```

Рисунок 7 – работа процесса loop

Вывод информации о процессах происходил примерно каждые 10 секунд. Мы можем видеть, что по мере течения времени процесс наращивал забираемые ресурсы компьютера.

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

```
danibrogue@jorgserver:~$ kill -15 1028
danibrogue@jorgserver:~$ ps -f
UID      PID  PPID  C STIME TTY          TIME CMD
danibro+    913    638  0 11:43 tty1        00:00:00 -bash
danibro+   1039    913  0 12:11 tty1      00:00:00 ps -f
[1]+  Terminated                  sh loop.txt
danibrogue@jorgserver:~$
```

Рисунок 8 – завершение процесса loop

После того, как мы дали команду завершить процесс loop, мы получили сообщение “Terminated” и название процесса.

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

```
danibrogue@jorgserver:~$ sh loop.txt&
[1] 1054
danibrogue@jorgserver:~$ ps -f
UID      PID  PPID  C STIME TTY          TIME CMD
danibro+  913    638  0 11:43  tty1        00:00:00  -bash
danibro+  1054    913  99 12:20  tty1        00:00:05 sh loop.txt
danibro+  1055    913  0 12:20  tty1        00:00:00 ps -f
danibrogue@jorgserver:~$ kill -9 1054
danibrogue@jorgserver:~$ ps -f
UID      PID  PPID  C STIME TTY          TIME CMD
danibro+  913    638  0 11:43  tty1        00:00:00  -bash
danibro+  1056    913  0 12:21  tty1        00:00:00 ps -f
[1]+  Killed                  sh loop.txt
danibrogue@jorgserver:~$
```

Рисунок 9 – запуск процесса в фоне и убийство

Запустить еще один экземпляр оболочки: bash.

```
danibrogue@jorgserver:~$ bash
danibrogue@jorgserver:~$ ps -f
UID      PID  PPID  C STIME TTY          TIME CMD
danibro+  913    638  0 11:43  tty1        00:00:00  -bash
danibro+  1068    913  0 12:27  tty1        00:00:00 bash
danibro+  1074   1068  0 12:27  tty1        00:00:00 ps -f
danibrogue@jorgserver:~$ _
```

Рисунок 10 – запуск дополнительной оболочки bash

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

```
danibrogue@jorgserver:~$ sh loop.txt&
[1] 926
danibrogue@jorgserver:~$ sh loop.txt&
[2] 927
danibrogue@jorgserver:~$ sh loop.txt&
[3] 928
danibrogue@jorgserver:~$ ps -f
UID      PID  PPID  C STIME TTY          TIME CMD
danibrogue+ 910    642  0 16:41 tty1    00:00:00 -bash
danibrogue+ 919    910  0 16:41 tty1    00:00:00 bash
danibrogue+ 926    919  48 16:41 tty1   00:00:03 sh loop.txt
danibrogue+ 927    919  35 16:41 tty1   00:00:02 sh loop.txt
danibrogue+ 928    919  30 16:41 tty1   00:00:01 sh loop.txt
danibrogue+ 929    919  0 16:41 tty1   00:00:00 ps -f
danibrogue@jorgserver:~$ kill -19 926
danibrogue@jorgserver:~$ kill -19 927

[1]+  Stopped                  sh loop.txt
danibrogue@jorgserver:~$ kill -19 928

[2]+  Stopped                  sh loop.txt
danibrogue@jorgserver:~$ ps -f
UID      PID  PPID  C STIME TTY          TIME CMD
danibrogue+ 910    642  0 16:41 tty1    00:00:00 -bash
danibrogue+ 919    910  0 16:41 tty1    00:00:00 bash
danibrogue+ 926    919  22 16:41 tty1   00:00:11 sh loop.txt
danibrogue+ 927    919  27 16:41 tty1   00:00:14 sh loop.txt
danibrogue+ 928    919  35 16:41 tty1   00:00:18 sh loop.txt
danibrogue+ 933    919  0 16:42 tty1   00:00:00 ps -f

[3]+  Stopped                  sh loop.txt
danibrogue@jorgserver:~$ _
```

Рисунок 11 – запуск и остановка процессов

```
danibrogue@jorgserver:~$ kill -18 926
danibrogue@jorgserver:~$ kill -18 927
danibrogue@jorgserver:~$ kill -18 928
danibrogue@jorgserver:~$ ps -f
UID      PID  PPID  C STIME TTY          TIME CMD
danibrogue+ 910    642  0 16:41 tty1    00:00:00 -bash
danibrogue+ 919    910  0 16:41 tty1    00:00:00 bash
danibrogue+ 926    919  13 16:41 tty1   00:00:18 sh loop.txt
danibrogue+ 927    919  12 16:41 tty1   00:00:17 sh loop.txt
danibrogue+ 928    919  14 16:41 tty1   00:00:19 sh loop.txt
danibrogue+ 934    919  0 16:43 tty1   00:00:00 ps -f
danibrogue@jorgserver:~$ _
```

Рисунок 12 – возобновление процессов

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

```
danibrogue@jorgserver:~$ sh loop.txt&
[1] 946
danibrogue@jorgserver:~$ sh loop.txt; sh loop.txt
^Z
[2]+  Stopped                  sh loop.txt
^Z
[3]+  Stopped                  sh loop.txt
danibrogue@Jorgserver:~$ ps -f
UID      PID  PPID  C STIME TTY      TIME CMD
danibrogue+ 930    644  0 09:48 tty1    00:00:00 -bash
danibrogue+ 939    930  0 09:48 tty1    00:00:00 bash
danibrogue+ 946    939  68 09:49 tty1   00:00:46 sh loop.txt
danibrogue+ 947    939  36 09:50 tty1   00:00:16 sh loop.txt
danibrogue+ 948    939  39 09:50 tty1   00:00:04 sh loop.txt
danibrogue+ 951    939  0 09:50 tty1   00:00:00 ps -f
danibrogue@jorgserver:~$ ps -f
UID      PID  PPID  C STIME TTY      TIME CMD
danibrogue+ 930    644  0 09:48 tty1    00:00:00 -bash
danibrogue+ 939    930  0 09:48 tty1    00:00:00 bash
danibrogue+ 946    939  74 09:49 tty1   00:01:02 sh loop.txt
danibrogue+ 947    939  26 09:50 tty1   00:00:16 sh loop.txt
danibrogue+ 948    939  16 09:50 tty1   00:00:04 sh loop.txt
danibrogue+ 955    939  0 09:51 tty1   00:00:00 ps -f
```

Рисунок 13 – работа с процессами

Команды:

1. sh loop.txt&
2. sh loop.txt; sh loop.txt
3. Ctrl+Z
4. Ctrl+Z
5. ps -f
6. ps -f

Как мы можем видеть по времени работы процессов, второй запускается только после остановки первого.

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

```
danibrogue@jorgserver:~$ bg  
[3]+ sh loop.txt &  
danibrogue@jorgserver:~$ ps -f  
UID      PID      PPID      C      STIME     TTY          TIME CMD  
danibrogue+ 930      644      0 09:48  tty1      00:00:00  -bash  
danibrogue+ 939      930      0 09:48  tty1      00:00:00  bash  
danibrogue+ 946      939      93 09:49  tty1      00:06:24  sh loop.txt  
danibrogue+ 947      939      4 09:50  tty1      00:00:16  sh loop.txt  
danibrogue+ 948      939      2 09:50  tty1      00:00:07  sh loop.txt  
danibrogue+ 963      939      0 09:56  tty1      00:00:00  ps -f
```

Рисунок 14 – перевод процесса в фоновый режим

Команды:

1. bg
2. ps -f

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

```
danibrogue@jorgserver:~$ ps -f  
UID      PID      PPID      C      STIME     TTY          TIME CMD  
danibrogue+ 930      644      0 09:48  tty1      00:00:00  -bash  
danibrogue+ 939      930      0 09:48  tty1      00:00:00  bash  
danibrogue+ 946      939      82 09:49  tty1      00:07:39  sh loop.txt  
danibrogue+ 947      939      3 09:50  tty1      00:00:16  sh loop.txt  
danibrogue+ 948      939      16 09:50  tty1      00:01:22  sh loop.txt  
danibrogue+ 967      939      0 09:59  tty1      00:00:00  ps -f  
danibrogue@jorgserver:~$ fg  
sh loop.txt  
^Z  
[2]+  Stopped                  sh loop.txt  
danibrogue@jorgserver:~$ ps -f  
UID      PID      PPID      C      STIME     TTY          TIME CMD  
danibrogue+ 930      644      0 09:48  tty1      00:00:00  -bash  
danibrogue+ 939      930      0 09:48  tty1      00:00:00  bash  
danibrogue+ 946      939      81 09:49  tty1      00:07:45  sh loop.txt  
danibrogue+ 947      939      3 09:50  tty1      00:00:19  sh loop.txt  
danibrogue+ 948      939      17 09:50  tty1      00:01:28  sh loop.txt  
danibrogue+ 968      939      0 09:59  tty1      00:00:00  ps -f  
danibrogue@jorgserver:~$ bg  
[2]+ sh loop.txt &  
danibrogue@jorgserver:~$ ps -f  
UID      PID      PPID      C      STIME     TTY          TIME CMD  
danibrogue+ 930      644      0 09:48  tty1      00:00:00  -bash  
danibrogue+ 939      930      0 09:48  tty1      00:00:00  bash  
danibrogue+ 946      939      80 09:49  tty1      00:07:52  sh loop.txt  
danibrogue+ 947      939      3 09:50  tty1      00:00:21  sh loop.txt  
danibrogue+ 948      939      17 09:50  tty1      00:01:35  sh loop.txt  
danibrogue+ 969      939      0 09:59  tty1      00:00:00  ps -f  
danibrogue@jorgserver:~$ fg  
sh loop.txt
```

Рисунок 15 – перевод задач

Команды:

1. ps -f
2. fg

3. Ctrl+Z
4. ps -f
5. bg
6. ps -f
7. fg

Создание канала и работа с ним

```
danibrogue@jorgserver:~$ mkfifo pipe
danibrogue@jorgserver:~$ ls -R > pipe&
[2] 963
danibrogue@jorgserver:~$ cat pipe
.:
loop2.txt
loop.txt
pipe
[2]- Done                  ls --color=auto -R > pipe
danibrogue@jorgserver:~$ cat loop.txt > pipe&
[2] 965
danibrogue@jorgserver:~$ cat pipe
while true; do true; done
[2]- Done                  cat loop.txt > pipe
danibrogue@jorgserver:~$
```

Рисунок 16 – создание канала и работа с ним

Команды:

1. mkfifo pipe
2. ls -R > pipe&
3. cat pipe
4. cat loop.txt > pipe&
5. cat pipe

Отобразить информацию о процессах указанного пользователя в виде иерархии, вывод отсортировать по значениям PID.

Используем команду “ps -H”:

```
danibrogue@jorgserver:~$ ps -H
 PID TTY      TIME CMD
 926 pts/1    00:00:00 bash
 940 pts/1    00:00:00  bash
 953 pts/1    00:00:36    sh
 958 pts/1    00:00:00    ps
danibrogue@jorgserver:~$ _
```

Рисунок 17 – отображенная информация

Команда “ps -H” выводит процессы в виде иерархии.

С помощью сигнала SIGSTOP приостановить выполнение процесса, владельцем которого является текущий пользователь. Через несколько секунд возобновить выполнение процесса.

Команды:

1. ps -f
2. sh loop.txt&
3. ps -f
4. kill -19 1028
5. ps -f
6. kill -18 1028
7. ps -f

```
danibrogue@jorgserver:~$ ps -f
UID      PID  PPID  C STIME TTY          TIME CMD
danibrogue+  926    639  0 13:53 ttys000  00:00:00 -bash
danibrogue+  940    926  0 13:55 ttys000  00:00:00 bash
danibrogue+ 1026    940  0 14:20 ttys000  00:00:00 ps -f
danibrogue@jorgserver:~$ sh loop.txt&
[1] 1028
danibrogue@jorgserver:~$ ps -f
UID      PID  PPID  C STIME TTY          TIME CMD
danibrogue+  926    639  0 13:53 ttys000  00:00:00 -bash
danibrogue+  940    926  0 13:55 ttys000  00:00:00 bash
danibrogue+ 1028    940  95 14:20 ttys000  00:00:16 sh loop.txt
danibrogue+ 1029    940  0 14:21 ttys000  00:00:00 ps -f
danibrogue@jorgserver:~$ kill -19 1028
danibrogue@jorgserver:~$ ps -f
UID      PID  PPID  C STIME TTY          TIME CMD
danibrogue+  926    639  0 13:53 ttys000  00:00:00 -bash
danibrogue+  940    926  0 13:55 ttys000  00:00:00 bash
danibrogue+ 1028    940  91 14:20 ttys000  00:00:39 sh loop.txt
danibrogue+ 1030    940  0 14:21 ttys000  00:00:00 ps -f

[1]+  Stopped                  sh loop.txt
danibrogue@jorgserver:~$ kill -18 1028
danibrogue@jorgserver:~$ ps -f
UID      PID  PPID  C STIME TTY          TIME CMD
danibrogue+  926    639  0 13:53 ttys000  00:00:00 -bash
danibrogue+  940    926  0 13:55 ttys000  00:00:00 bash
danibrogue+ 1028    940  69 14:20 ttys000  00:00:42 sh loop.txt
danibrogue+ 1031    940  0 14:21 ttys000  00:00:00 ps -f
```

Рисунок 18 – остановка и возобновление процесса

Число 19 в команде kill отвечает за приостановку процесса, число 18 – за возобновление

Определить идентификаторы и имена процессов, не связанных с указанным терминалом.

Используем команду “ps T -N”:

```
261 ? I< 0:00 [ext4-rsv-conver]
329 ? S<s 0:00 /lib/systemd/systemd-journald
357 ? I 0:00 [kworker/0:4-events]
360 ? Ss 0:01 /lib/systemd/systemd-udevd
372 ? I< 0:00 [iprt-VBoxWQueue]
504 ? I< 0:00 [kaluad]
505 ? I< 0:00 [kmpath_rdacd]
506 ? I< 0:00 [kmpathd]
507 ? I< 0:00 [kmpath_handlerd]
508 ? SLs1 0:00 /sbin/multipathd -d -s
518 ? S< 0:00 [loop0]
520 ? S< 0:00 [loop1]
522 ? S< 0:00 [loop2]
523 ? S< 0:00 [loop3]
526 ? S< 0:00 [loop4]
527 ? S< 0:00 [loop5]
528 ? S 0:00 [jbd2/sda2-8]
529 ? I< 0:00 [ext4-rsv-conver]
537 ? Ss1 0:00 /lib/systemd/systemd-timesyncd
587 ? Ss 0:00 /lib/systemd/systemd-networkd
594 ? Ss 0:00 /lib/systemd/systemd-resolved
608 ? Ss1 0:00 /usr/lib/accountsservice/accounts-daemon
611 ? Ss 0:00 /usr/sbin/cron -f
612 ? Ss 0:00 /usr/bin/dbus-daemon --system --address=systemd: --nofork --nopidfile -
623 ? Ss 0:00 /usr/bin/python3 /usr/bin/networkd-dispatcher --run-startup-triggers
624 ? Ss1 0:00 /usr/sbin/rsyslogd -n -iNONE
626 ? Ss1 0:01 /usr/lib/snapd/snapd
634 ? Ss 0:00 /lib/systemd/systemd-logind
637 ? Ss 0:00 /usr/sbin/atd -f
659 ? Ss1 0:00 /usr/bin/python3 /usr/share/unattended-upgrades/unattended-upgrade-shut
665 ? Ss 0:00 sshd: /usr/sbin/sshd -D [listener] 0 of 10-100 startups
704 ? Ss1 0:00 /usr/lib/polkit-1/polkitd --no-debug
787 ? I 0:00 [kworker/u2:2-events_power_efficient]
788 ? I 0:00 [kworker/0:1-events]
912 ? Ss 0:00 /lib/systemd/systemd --user
913 ? S 0:00 (sd-pam)
danibrogue@jorgserver:~$
```

Рисунок 19 – процессы, не связанные с терминалом

Команда “ps T” выводит процессы, связанные с текущим терминалом. В свою очередь аргумент “-N” инвертирует показания, т.е. заставляет команду вывести процессы, не связанные с текущим терминалом.

Вывод

Ознакомились на практике с понятием процесса в операционной системе. Приобрели опыт и навыки управления процессами в операционной системе Linux.