Липецкий государственный технический университет

Факультет автоматизации и информатики

Кафедра Автоматизированных систем управления

ЛАБОРАТОРНАЯ РАБОТА №3

по дисциплине «Операционная система Linux»

Процессы в операционной системе Linux

Студентка Жидков И.А.

Группа АС-19

Руководитель Кургасов В.В.

к.п.н.

Липецк 2021

Оглавление

Цель работы

Ознакомиться на практике с понятием процесса в операционной системе.

Приобрести опыт и навыки управления процессами в операционной системе Linux.

Задание кафедры

Вариант 3.

1. Сгенерировать следующую информацию о m (m>2) процессах системы, имеющих значение идентификатора больше заданного n: флаг — сведения о процессе, статус, PID, PPID, приоритет, использованное время и имя программы.

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

3. Через символ « : » вывести идентификаторы процессов, для которых родителем является командный интерпретатор.

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.

5. Вывести общую информацию о системе

5.1 Вывести информацию о текущем интерпретаторе команд

5.2 Вывести информацию о текущем пользователе

5.3 Вывести информацию о текущем каталоге

5.4 Вывести информацию об оперативной памяти и области подкачки

5.5 Вывести информацию о дисковой памяти

6.Выполнить команды получения информации о процессах

6.1 Получить идентификатор текущего процесса(PID)

6.2 Получить идентификатор родительского процесса(PPID)

6.3 Получить идентификатор процесса инициализации системы

6.4 Получить информацию о выполняющихся процессах текущего пользователя в текущем интерпретаторе команд

6.5 Отобразить все процессы

7. Выполнить команды управления процессами

7.1 Получить информацию о выполняющихся процессах текущего пользователя в текущем интерпретаторе

7.2 Определить текущее значение nice по умолчанию

7.2 Запустить интерпретатор bash с понижением приоритета

nice -n 10 bash

7.3 Определить PID запущенного интерпретатора

7.4 Установить приоритет запущенного интерпретатора равным 5

renice -n 5 <PID процесса>

7.5 Получить информацию о процессах bash

ps lax | grep bash

Ход работы

Запустим виртуальную машину Linux Ubuntu. Выведем информацию о состоянии процессов системы при помощи команды ps --sort rss. Таким образом, список будет выведен в порядке убывания PID

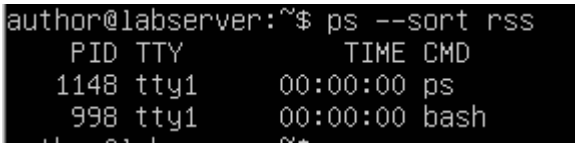

Рисунок 1 – Вывод с сортировкой по убыванию

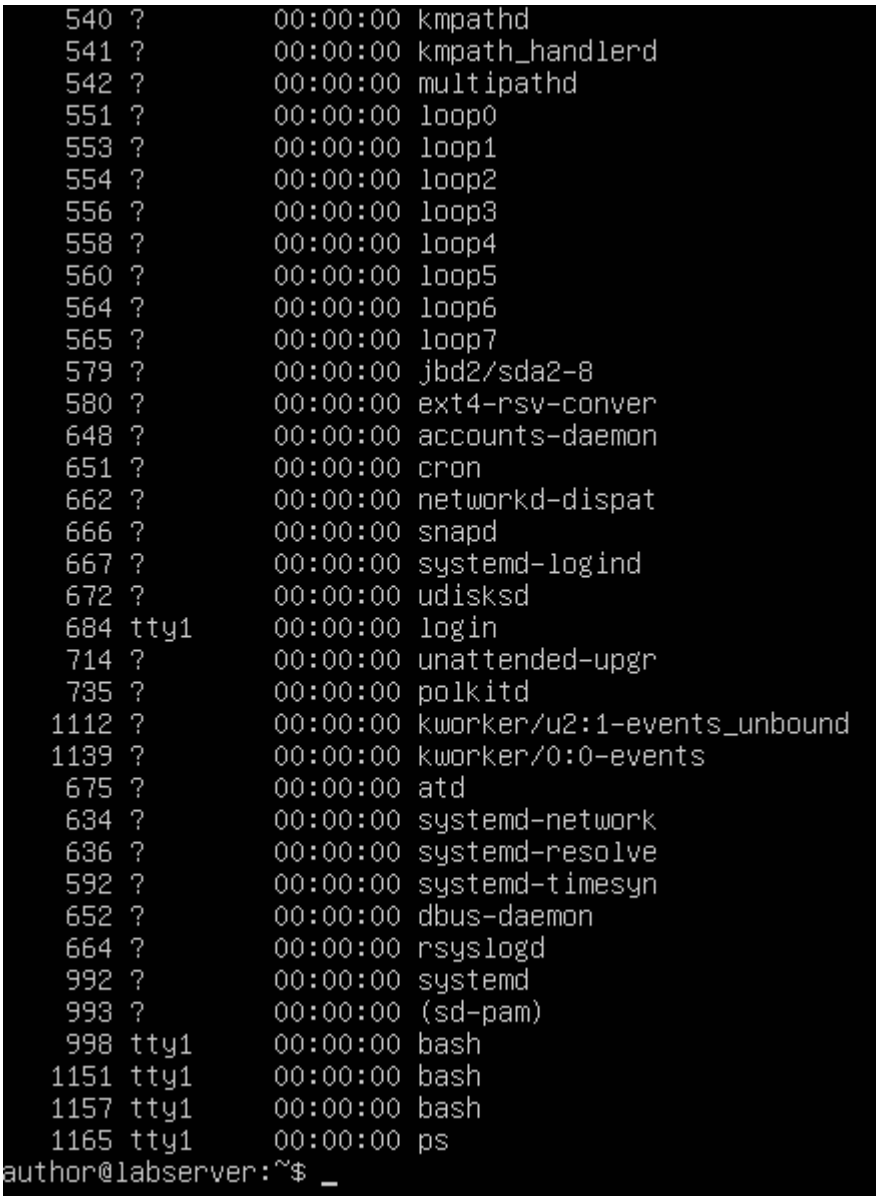

Рисунок 2 – Вывод всех процессов по возрастанию

Завершим выполнение двух процессов, владельцем которых является текущий пользователь. Первый процесс завершаем с помощью сигнала SIGKILL, задав его имя, второй — с помощью сигнала SIGINT, задав его номер.

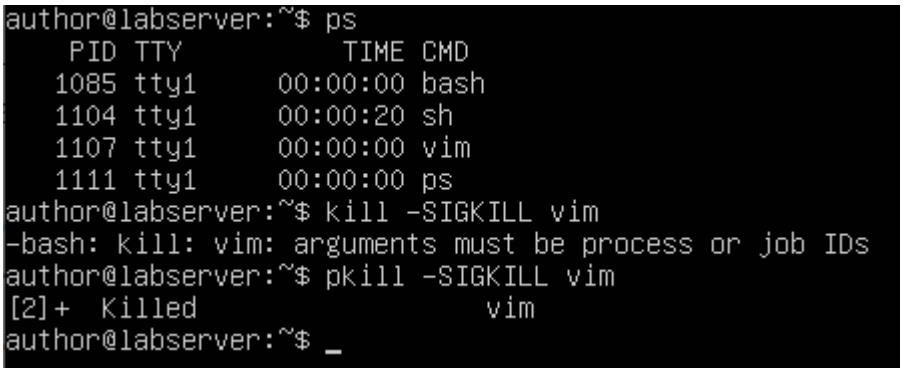

Рисунок 3 – Завершение процесса с помощью pkill

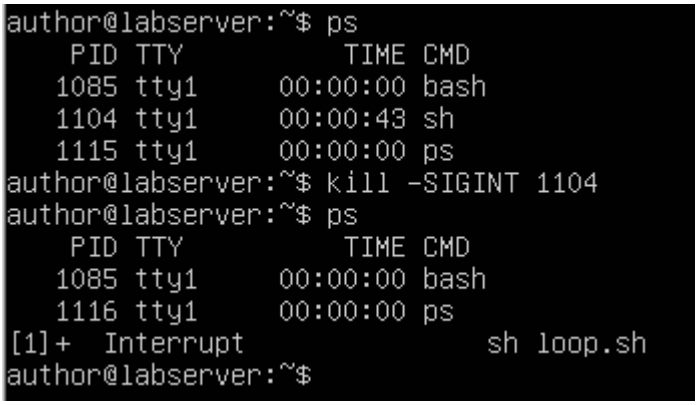

Рисунок 4 – Завершение процесса с помощью сигнала SIGINT

вывести идентификаторы процессов, для которых родителем является командный интерпретатор.

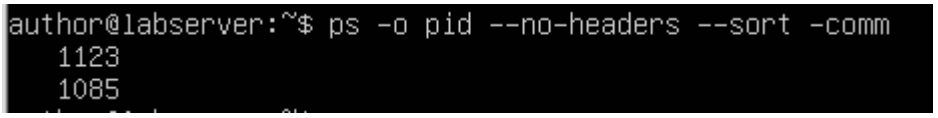

Рисунок 5 – Вывод идентификаторов процессов, запущенных пользователем

Выведем информацию о текущем интерпретаторе команд с помощью команды echo $SHELL и echo $BASH_VERSION. Переменная окружения SHELL хранит путь до исполняемого файла оболочки.

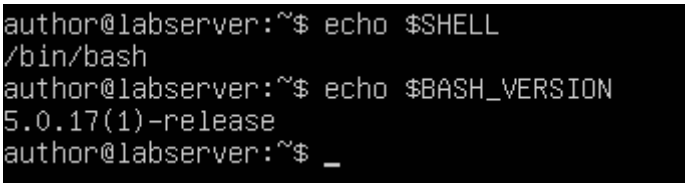

Рисунок 6 – Вывод информации о текущем интерпретаторе команд

Выведем информацию о текущем пользователе. Это можно сделать с помощью команды whoami.

Рисунок 7 – Вывод информации о текущем пользователе

Выведем информацию о текущем каталоге с помощью команд pwd и ls. Pwd показывает путь до текущего каталога, а ls выводит его содержимое.

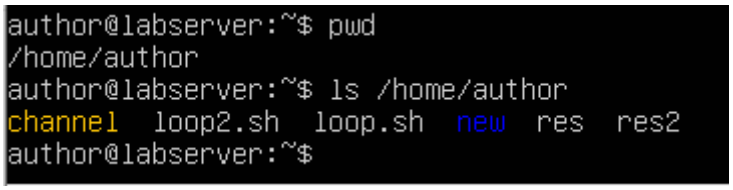

Рисунок 8 – Вывод информации о текущем каталоге

Выведем информацию об оперативной памяти и области подкачки с помощью команд free и top.

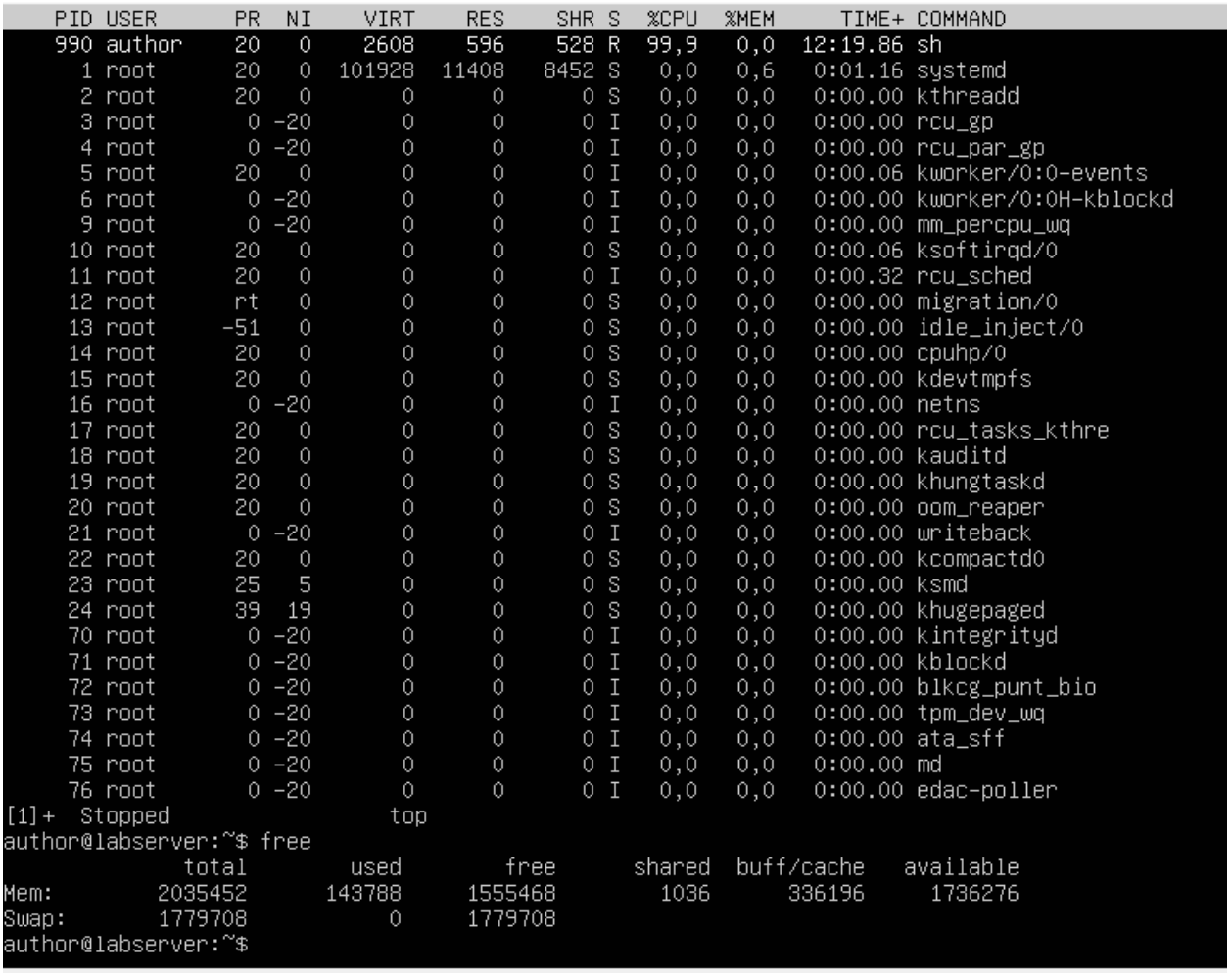

Рисунок 9 – Вывод информации об оперативной памяти и области подкачки с помощью команды free

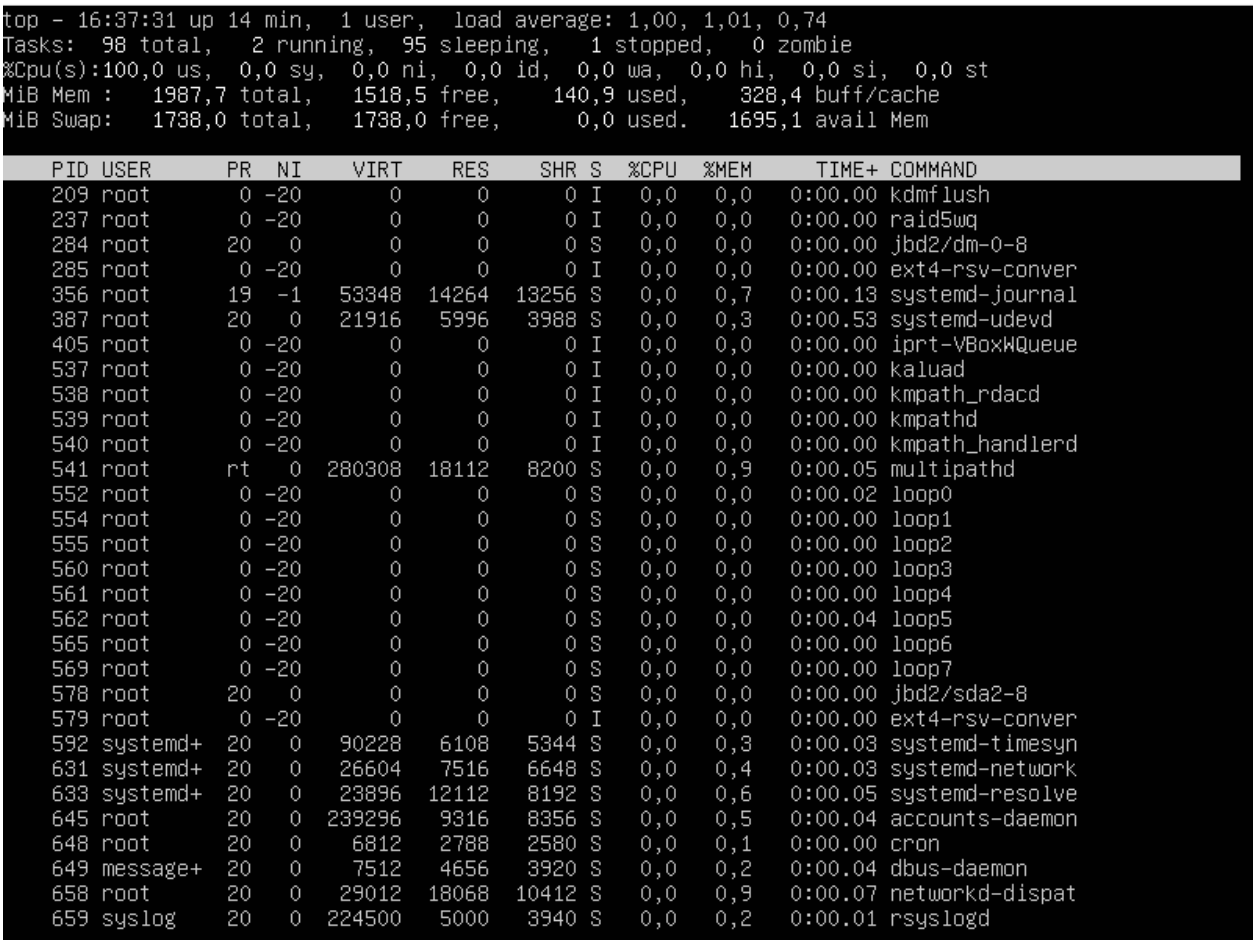

Рисунок 10 – Вывод информации об оперативной памяти и области подкачки с помощью команды top

Выведем информацию о дисковой памяти с помощью команд ls -l /dev/ и df.

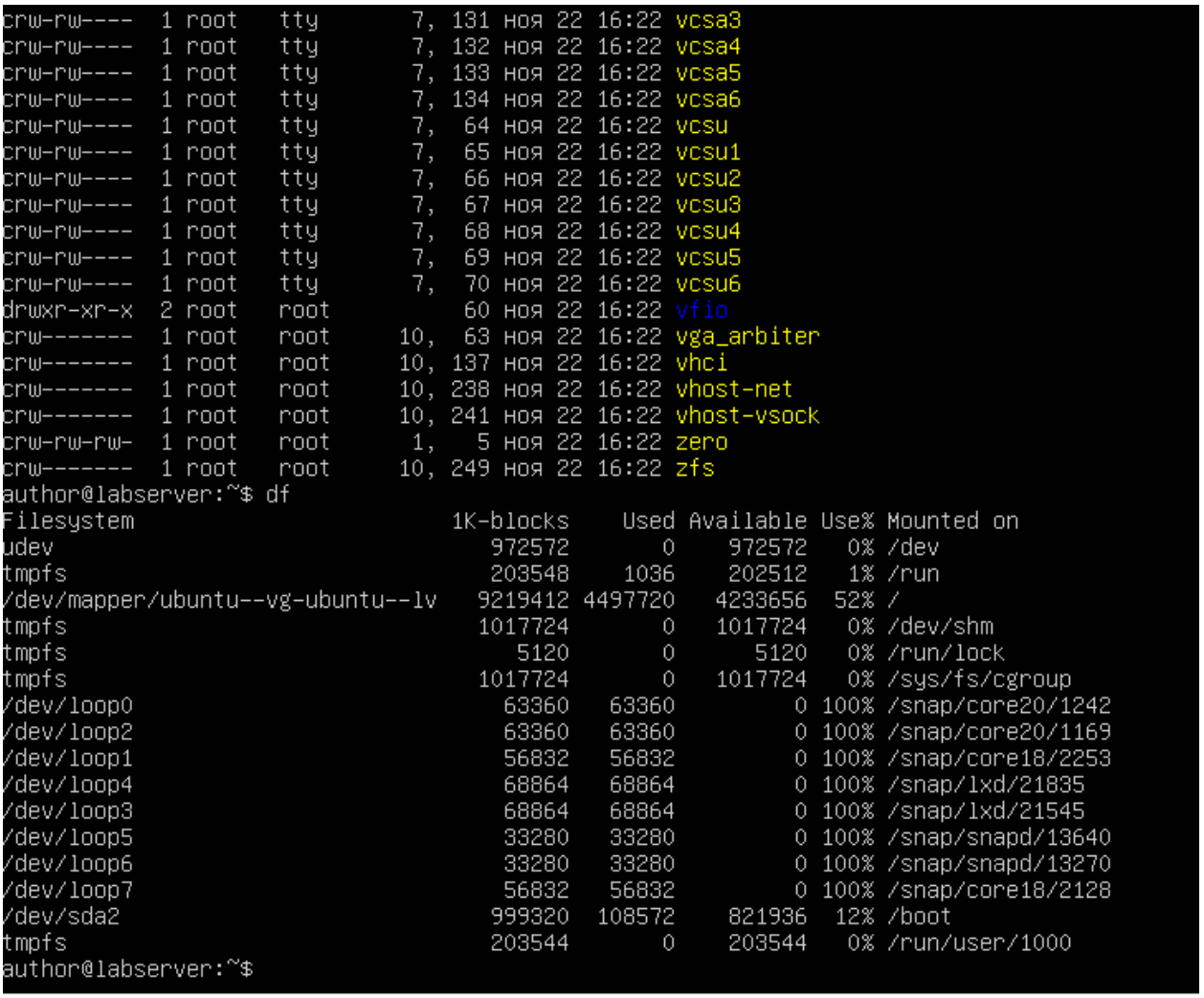

Рисунок 11 – Вывод информации о дисковой памяти с помощью команд ls -l /dev/ и df.

Получим идентификатор текущего процесса(PID). Самый распространённый способ узнать PID Linux - использовать команду ps, которая выведет все текущие процессы и их PID.

Рисунок 12 – Получение PID процесса

Получим идентификатор родительского процесса(PPID) с помощью ps –f.

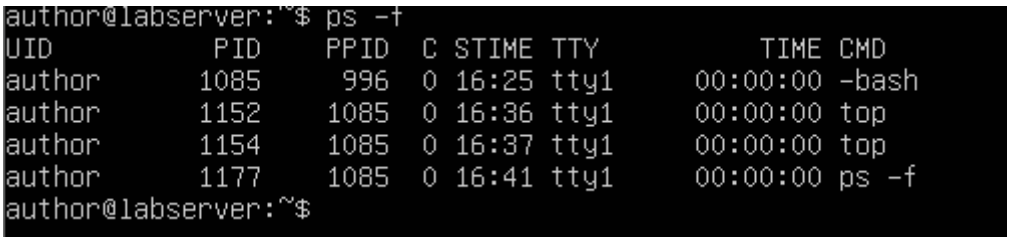

Рисунок 13 – Получение PPID

Получим идентификатор процесса инициализации системы. Поскольку этим процессом является init, то его PID можно легко узнать с помощью команды pidof имя_процесса.

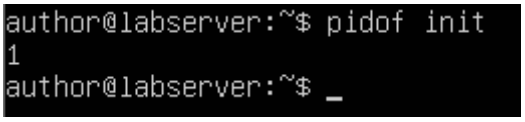

Рисунок 14 – Получение PID процесса инициализации системы

Получим информацию о выполняющихся процессах текущего пользователя в текущем интерпретаторе команд. Сделать это можно с помощью команды ps -a -f.

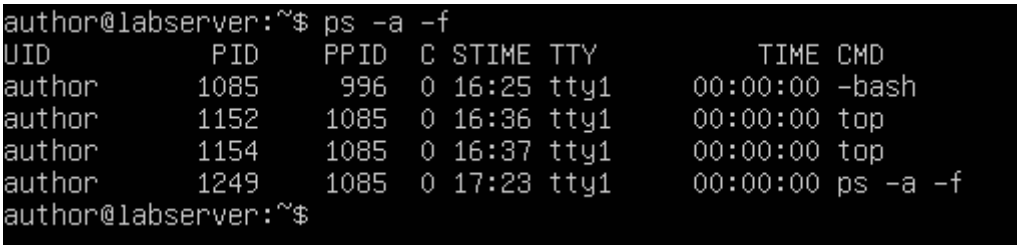

Рисунок 15 - Получение информации о выполняющихся процессах текущего пользователя в текущем интерпретаторе команд.

Чтобы отобразить все процессы используем ps -e.

```
    552 ?        00:00:00 loop0
    554 ?        00:00:00 loop1
    555 ?        00:00:00 loop2
    560 ?        00:00:00 loop3
    561 ?        00:00:00 loop4
    562 ?        00:00:00 loop5
    565 ?        00:00:00 loop6
    569 ?        00:00:00 loop7
    578 ?        00:00:00 jbd2/sda2-8
    579 ?        00:00:00 ext4-rsv-conver
    592 ?        00:00:00 systemd-timesyn
    631 ?        00:00:00 systemd-network
    633 ?        00:00:00 systemd-resolve
    645 ?        00:00:00 accounts-daemon
    648 ?        00:00:00 cron
    649 ?        00:00:00 dbus-daemon
    658 ?        00:00:00 networkd-dispat
    659 ?        00:00:00 rsyslogd
    662 ?        00:00:00 snapd
    663 ?        00:00:00 systemd-logind
    668 ?        00:00:00 udisksd
    669 ?        00:00:00 atd
    670 ?        00:00:01 kworker/0:4-events
    703 ?        00:00:00 unattended-upgr
    719 ?        00:00:00 polkitd
    969 ?        00:00:00 systemd
    971 ?        00:00:00 (sd-pam)
    990 ?        00:59:04 sh
    996 tty1     00:00:00 login
   1085 tty1     00:00:00 bash
   1152 tty1     00:00:00 top
   1154 tty1     00:00:00 top
   1225 ?        00:00:00 kworker/0:0-cgroup_destroy
   1231 ?        00:00:00 kworker/u2:2-events_power_efficient
   1235 ?        00:00:00 kworker/u2:0-events_unbound
   1250 tty1     00:00:00 ps
author@labserver:~$
```

Рисунок 16 – Отображение всех процессов.

Определим текущее значение nice по умолчанию. По умолчанию nice устанавливает значение приоритета 10.

```
author@labserver:~$ ps -l
F S   UID     PID    PPID  C PRI  NI ADDR SZ WCHAN  TTY          TIME CMD
4 S  1000    1085     996  0  80   0 -  2099 do_wai tty1     00:00:00 bash
0 T  1000    1152    1085  0  80   0 -  2285 do_sig tty1     00:00:00 top
0 T  1000    1154    1085  0  80   0 -  2285 do_sig tty1     00:00:00 top
0 R  1000    1251    1085  0  80   0 -  2203 -      tty1     00:00:00 ps
author@labserver:~$
```

Рисунок 17 – Определение nice

Запустим интерпретатор bash с понижением приоритета используя nice -n 10 bash. Для этого загрузимся как пользователь с root правами и посмотрим имеющиеся процессы.

```
author@labserver:~$ ps -l
F S   UID     PID    PPID  C PRI  NI ADDR SZ WCHAN  TTY          TIME CMD
4 S  1000    1085     996  0  80   0 -  2099 do_wai tty1     00:00:00 bash
0 T  1000    1152    1085  0  80   0 -  2285 do_sig tty1     00:00:00 top
0 T  1000    1154    1085  0  80   0 -  2285 do_sig tty1     00:00:00 top
0 R  1000    1251    1085  0  80   0 -  2203 -      tty1     00:00:00 ps
author@labserver:~$ nice -n 10 bash
```

Рисунок 18 – Запуск bash с понижением приоритета

Определим PID запущенного интерпретатора.

```
author@labserver:~$ ps -l
F S   UID     PID    PPID  C PRI  NI ADDR SZ WCHAN  TTY          TIME CMD
4 S  1000    1085     996  0  80   0 -  2099 do_wai tty1     00:00:00 bash
0 T  1000    1152    1085  0  80   0 -  2285 do_sig tty1     00:00:00 top
0 T  1000    1154    1085  0  80   0 -  2285 do_sig tty1     00:00:00 top
0 S  1000    1252    1085  0  90  10 -  2068 do_wai tty1     00:00:00 bash
0 R  1000    1259    1252  0  90  10 -  2203 -      tty1     00:00:00 ps
author@labserver:~$ _
```

Рисунок 19 – Определение PID bash

Установим приоритет запущенного интерпретатора равным 5 с помощью renice -n 5 <PID процесса>.

```
author@labserver:~$ renice -n 5 1085
1085 (process ID) old priority 0, new priority 5
author@labserver:~$ _
```

Рисунок 20 – Установка нового приоритета

Получение информации о процессах bash с помощью ps lax | grep bash.

```
author@labserver:~$ ps lax | grep bash
4  1000    1085     996  25   5   8396  5300 do_wai SN   tty1       0:00 -bash
0  1000    1252    1085  30  10   8272  5084 do_wai SN   tty1       0:00 bash
0  1000    1264    1252  30  10   6300   728 pipe_w SN+  tty1       0:00 grep --color=auto bash
author@labserver:~$
```

Рисунок 21 – Получение информации о процессах

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.

Контрольные вопросы

l. Перечислите состояния задачи в ОС Ubuntu.

Задача переходит в состояние running (выполнения) после выделения ей процессора. При блокировке задача переходит в состояние sleeping (спячки), а при остановке работы в состояние останов (stopped). Состояние zombie (зомби) показывает, что выполнение задачи прекратилось, однако она еще не была удалена из системы. Например, если процесс состоит из нескольких потоков, он будет пребывать в состоянии зомби, пока все потоки не получат уведомление о завершении работы основного процесса. Задача в состоянии dead (смерти) может быть удалена из системы. Состояния active (активный) и expired (неактивный) используются при планировании выполнения процесса, и поэтому они не сохраняются в переменной state.

2. Как создаются задачи в ОС Ubuntu?

Задачи создаются путем вызова функции clone.

3. Назовите классы потоков ОС Ubuntu

1.Потоки реального времени, обслуживаемые по алгоритму FIFO.

2.Потоки реального времени, обслуживаемые в порядке циклической очереди.

3.Потоки разделения времени

4. Как используется приоритет планирования при запуске задачи

Для Linux приоритет планирования процесса реального времени варьируется от 1 (самый низкий приоритет) до 99 (самый высокий приоритет). Обычные процессы также имеют приоритет планирования, равный 0.

5. Как можно изменить приоритет для выполняющейся задачи?

С помощью renice

renice — UNIX-утилита, позволяющая изменить приоритет запущенных задач. Привилегированный пользователь (root) может указать отрицательное смещение. Команда renice может смещать приоритет в диапазоне от -20 (наивысший приоритет) до 19 (низший приоритет) от текущего. Для изменения значения приоритета отдельных процессов достаточно перечислить их идентификаторы.