Липецкий государственный технический университет

Факультет автоматизации и информатики

Кафедра автоматизированных систем управления

ЛАБОРАТОРНАЯ РАБОТА №4

по Операционной системе Linux

Управление процессами в ОС Ubuntu

Студент Лобов М.Ю.

Группа АИ-18

Руководитель Кургасов В.В.

Доцент, к.п.н.

Липецк 2020 г.

Оглавление

Цель работы

Познакомиться со средствами управления процессами ОС Ubuntu.

Задание кафедры

1. Запустить программу виртуализации Oracle VM VirtualBox.
2. Запустить виртуальную машину Ubuntu.
3. Открыть окно интерпретатора команд.
4. Вывести общую информацию о системе.
   4.1. Вывести информацию о текущем интерпретаторе команд.
   4.2. Вывести информацию о текущем пользователе.
   4.3. Вывести информацию о текущем каталоге.
   4.4. Вывести информацию об оперативной памяти и области подкачки.
   4.5. Вывести информацию о дисковой памяти.
5. Выполнить команды получения информации о процессах.
   5.1. Получить идентификатор текущего процесса (PID).
   5.2. Получить идентификатор родительского процесса (PPID).
   5.3. Получить идентификатор процесса инициализации системы.
   5.4. Получить информацию о выполняющихся процессах текущего пользователя в текущем интерпретаторе команд.
   5.5. Отобразить все процессы.
6. Выполнить команды управления процессами.
   6.1. Получить информацию о выполняющихся процессах текущего пользователя в текущем интерпретаторе.
   6.2. Определить текущее значение **nice** по умолчанию.
   6.3. Запустить интерпретатор **bash** с понижением приоритета.
   6.4. Определить PID запущенного интерпретатора.
   6.5. Установить приоритет запущенного интерпретатора равным 5.
   6.6. Получить информацию о процессах **bash**.

Ход работы

В связи с тем, что на рабочей машине установлена ОС Linux Ubuntu, а не Linux Ubuntu Server, было принято решение выполнить лабораторную работу в установленном дистрибутиве. Серьёзных отличий не выявлено.

Первым делом выведем информацию о текущем интерпретаторе команд. Сделаем это с помощью команды **echo $SHELL**:

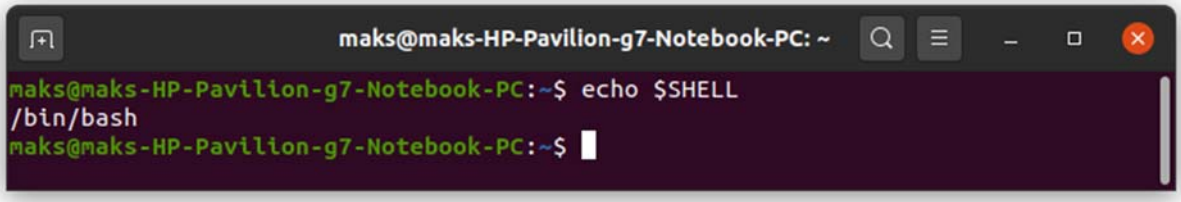

Рисунок 1 – Информация о текущем интерпретаторе команд

Далее выполним похожее задание: теперь выведем информацию о текущем пользователе, а именно его имя. Сделаем это с помощью команды **whoami**:

Рисунок 2 – Информация о текущем пользователе

Теперь выведем информацию о текущем каталоге (команда **pwd**):

Рисунок 3 – Информация о текущем пользователе

Следующим шагом будет вывод информации об оперативной памяти и области подкачки. В этом нам поможет команда **free**:

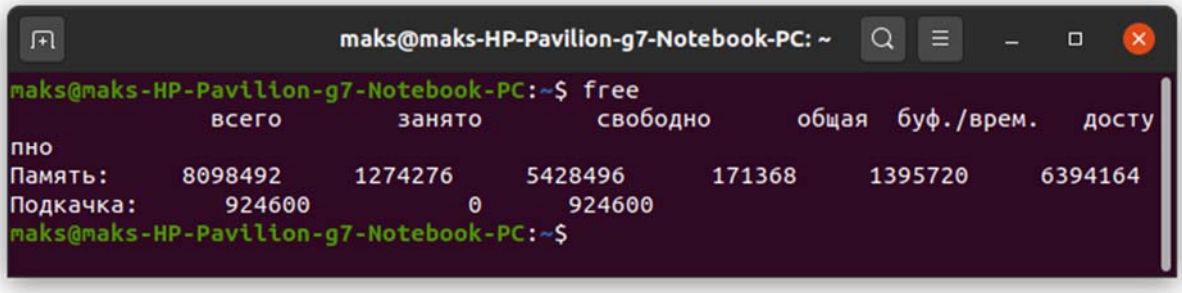

Рисунок 4 – Информация об оперативной памяти и области подкачки

Последним заданием в разделе будет вывод информации о дисковой памяти. Для этого задействуем команду **df**:

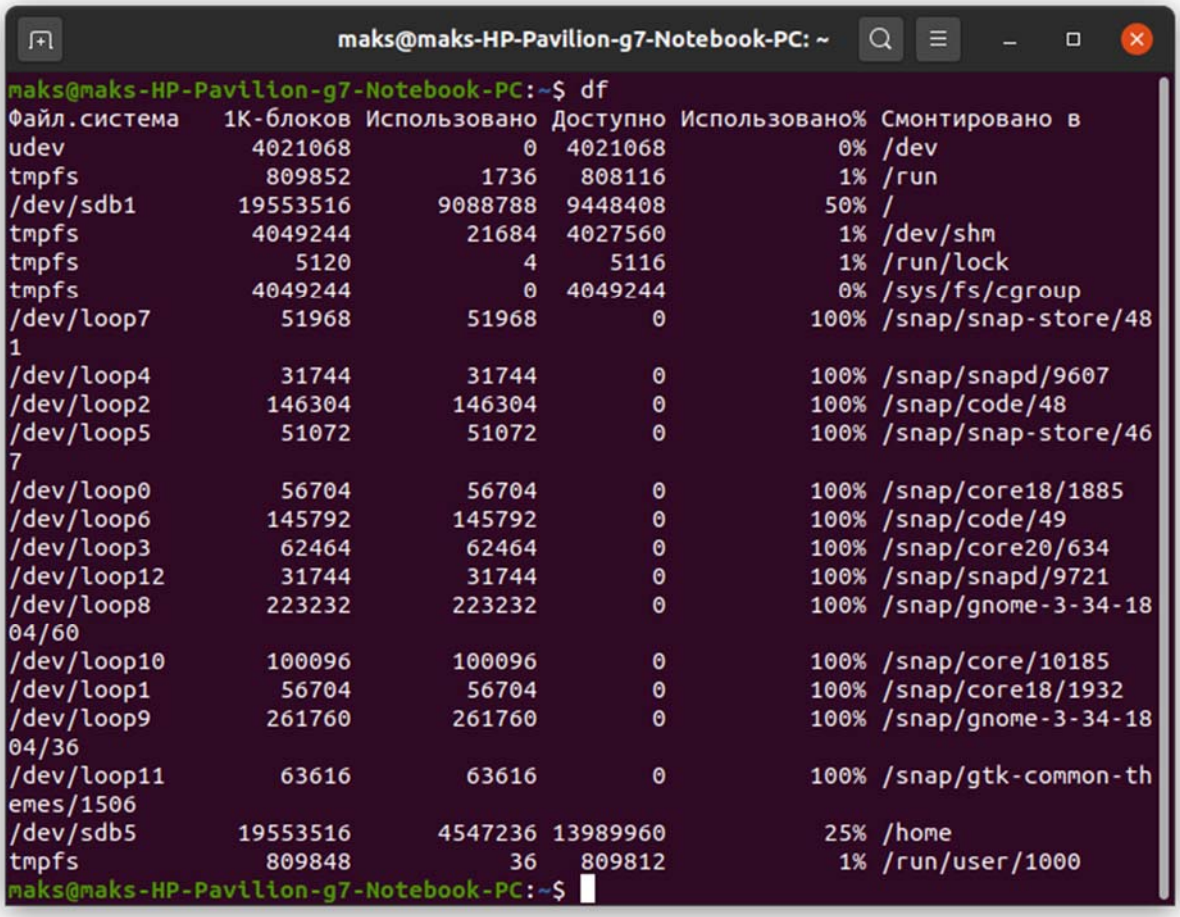

Рисунок 5 – Информация о дисковой памяти

Теперь перейдём к выполнению заданий из раздела вывода информации о процессах.

Первым заданием будет получение идентификатора текущего процесса (PID). Сделаем это с помощью команды **echo $$**:

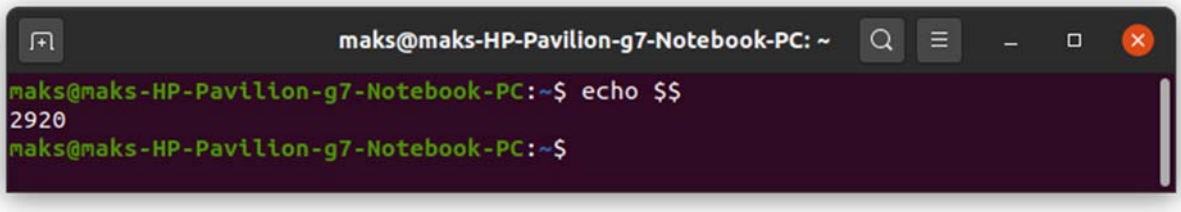

Рисунок 6 – Идентификатор текущего процесса

Следующим шагом получим идентификатор родительского процесса (PPID). Используем команду **echo $PPID**:

Рисунок 7 – Идентификатор родительского процесса

Для выполнения следующего задания, а именно получения идентификатора процесса инициализации системы, нам понадобится команда **pidof**, которая выводит идентификатор процесса по названию процесса:

Рисунок 8 – Идентификатор процесса инициализации системы

Теперь выведем информацию о выполняющихся процессах текущего пользователя в текущем интерпретаторе команд. Для этого нам нужно исполнить команду **ps** без аргументов:

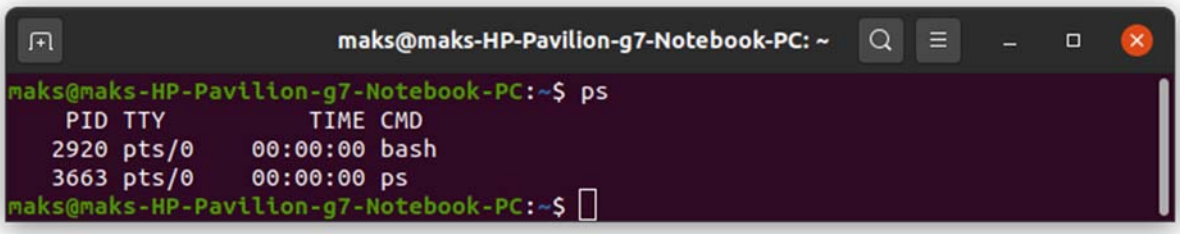

Рисунок 9 – Процессы текущего пользователя в текущем интерпретаторе

Если же нам нужно вывести вообще все процессы, то нам нужно выполнить команду **ps** с аргументом -e:

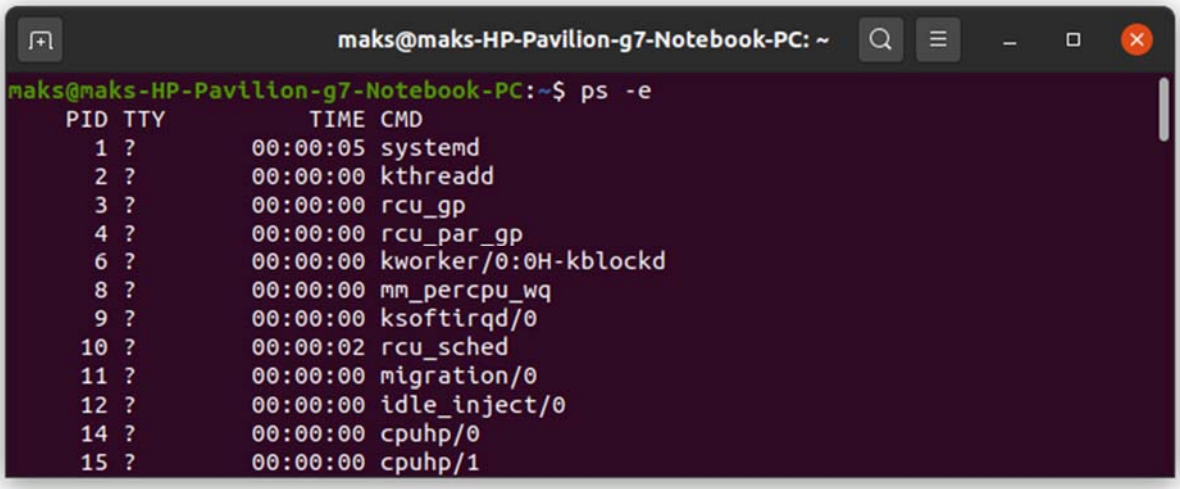

Рисунок 10 – Все процессы

В последнем разделе нам придётся познакомиться с командами управления процессами.

Выполним следующие действия: выведем информацию о выполняющихся процессах текущего пользователя в текущем интерпретаторе команд (**ps**) и определим текущее значение **nice** по умолчанию (**nice**). Затем запустим новый экземпляр интерпретатора **bash** с понижением приоритета (**nice –n 10 bash**) и узнаем его идентификатор (**echo $$**). Затем установим приоритет запущенного интерпретатора равным 5 (**renice –n 5 1977**) и получим информацию о процессах **bash** (**ps lax | grep bash**). Сделаем это:

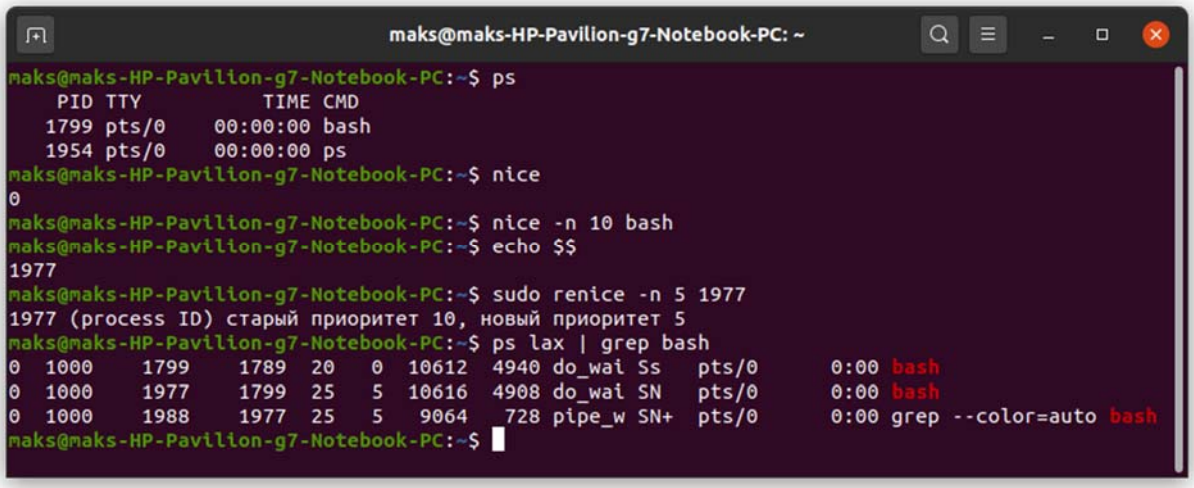

Рисунок 11 – Операции с созданным интерпретатором

Вывод

В ходе выполнения лабораторной работы мы познакомились со средствами управления процессами в ОС Ubuntu и изучили некоторые команды, способные помочь в работе с процессами в Linux.

Ответы на контрольные вопросы

1. Перечислите состояния задачи в ОС Ubuntu.

1) Running (выполнение) – после выделения ей процесса;
2) Sleeping (спячки) – состояние блокировки;
3) Stopped (останов) – остановка работы;
4) Zombie (зомби) – выполнение задачи прекратилась, однако она не была удалена;
5) Dead (смерть) – задача может быть удалена из системы;
6) Active (активный) и inspired (неактивный) – используются при планировании выполнения процесса.

2. Как создаются задачи в ОС Ubuntu?

В системе Linux и процессы, и потоки называют задачами (**task**). Изнутри они представляют собой единую структуру данных.

Планировщик процессов хранит список всех задач в виде двух структур данных.

Первая структура представляет собой кольцевой список, каждая запись которого содержит указатели на предыдущую и последующую задачу. Обращение к этой структуре происходит в том случае, когда ядру необходимо проанализировать все задачи, которые должны быть выполнены в системе.

Второй структурой является хэш-таблица. При создании задачи ей присваивается уникальный идентификатор процесса (**process identifier, PID**). Идентификаторы процессов передаются хэш-функции для определения местоположения процесса в таблице процессов. Хэш-метод обеспечивает быстрый доступ к специфическим структурам данных задачи, если ядру известен ее PID.

Каждая задача таблицы процессов представляется в виде структуры **task_struct**, служащей в роли дескриптора процесса (т.е. блока управления процессором (PCB). В структуре **task_struct** хранятся переменные и вложенные структуры, описывающие процесс.

3. Назовите классы потоков в 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?

Планировщик использует приоритет и квант следующим образом. Сначала он вычисляет называемую в системе Linux «добродетелью» (**goodness**) величину каждого готового процесса по следующему алгоритму:

if (class == real_time) goodness = 1000 + priority;

if (class == timesharing & quantum > 0) goodness = quantum + priority;

if (class == timesharing && quantum == 0) goodness = 0;

В остальном алгоритм планирования очень прост: когда нужно принять решение, выбирается поток с максимальным значением «добродетели». Во время работы процесса его квант (переменная **quantum**) уменьшается на единицу на каждом тике. Центральный процессор отнимается у потока при выполнении одного из следующих условий:

1. **Квант потока уменьшился до 0.**
2. **Поток блокируется на операции ввода-вывода и т.д.**
3. **В состояние готовности перешел ранее заблокированный поток с более высокой «добродетелью».**

5. Как можно изменить приоритет для выполняющейся задачи?

С этим может помочь команда renice –n <**новое значение приоритета**> <**PID запущенного процесса**>