

Липецкий государственный технический университет

Факультет автоматизации и информатики

Кафедра Автоматизированных систем управления

ЛАБОРАТОРНАЯ РАБОТА №4

По дисциплине «ОС Linux»

Управление процессами ОС Ubuntu

Студент

Печенкин Д.В.

Группа ПИ-18

Руководитель

Кургасов В.В.

Доцент

Липецк 2020г

1. Цель работы

Ознакомиться со средствами управления процессами ОС Ubuntu.

2. Задание кафедры

- 1) Запустить программу виртуализации Oracle VM VirtualBox, запустить виртуальную машину ubuntu и открыть окно интерпретатора команд.
- 2) Вывести общую информацию о системе
 - 2.1) Вывести информацию о текущем интерпретаторе команд
 - 2.3) Вывести информацию о текущем пользователе
 - 2.3) Вывести информацию о текущем каталоге
 - 2.4) Вывести информацию об оперативной памяти и области подкачки
 - 2.5) Вывести информацию о дисковой памяти
- 3) Выполнить команды получения информации о процессах
 - 3.1) Получить идентификатор текущего процесса (PID)
 - 3.2) Получить идентификатор родительского процесса (PPID)
 - 3.3) Получить идентификатор процесса инициализации системы
 - 3.4) Получить информацию о выполняющихся процессах текущего пользователя в текущем интерпретаторе команд
 - 3.5) Отобразить все процессы
- 4) Выполнить команды управления процессами
 - 4.1) Получить информацию о выполняющихся процессах текущего пользователя в текущем интерпретаторе
 - 4.2) Определить текущее значение **nice** по умолчанию
 - 4.3) Запустить интерпретатор bash с понижением приоритета nice -n 10
bash
 - 4.4) Определить PID запущенного интерпретатора
 - 4.5) Установить приоритет запущенного интерпретатора равным 5
Renice -n 5 <PID процесса>
 - 4.6) Получить информацию о процессах bash
Ps lax| grep bash

3. Выполнение работы

3.1 Запуск Ubuntu

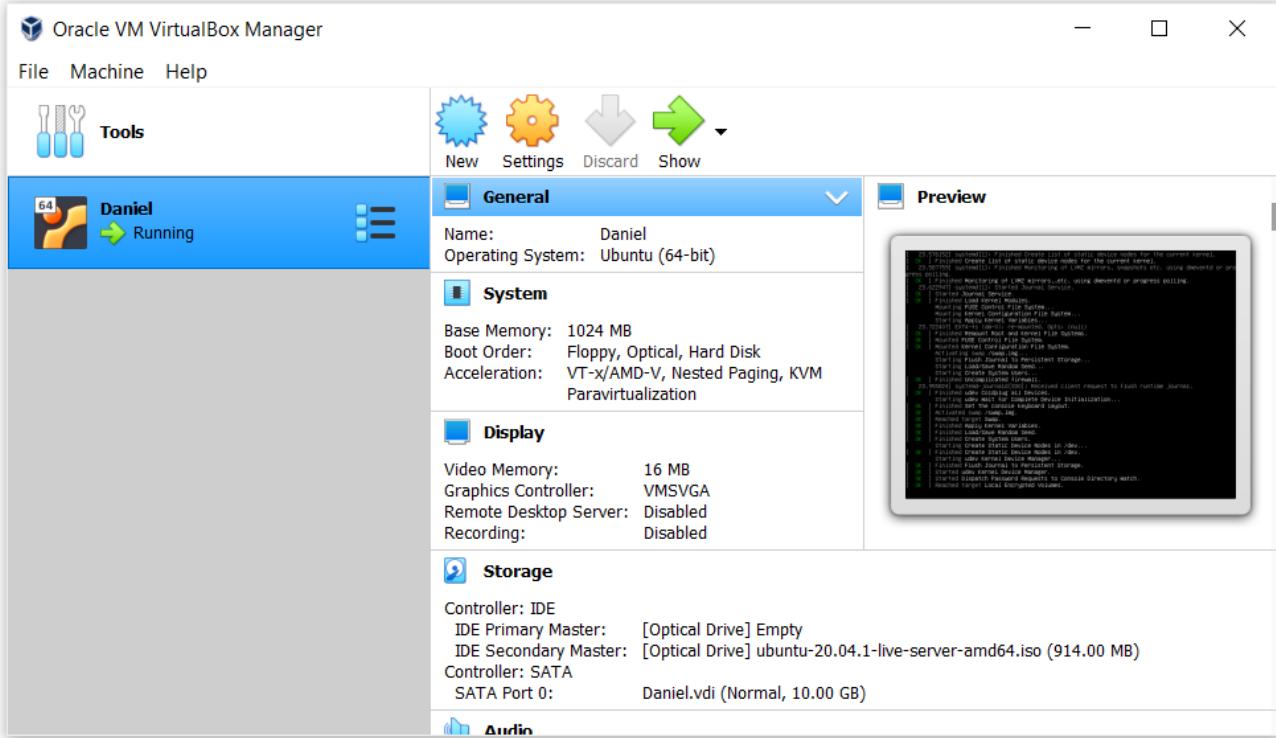


Рисунок 1 – Запуск виртуальной машины

После запуска, необходимо нажать на иконку “Запустить”, для запуска Ubuntu.

После запуска ОС, необходимо ввести логин и пароль для входа

```
Ubuntu 20.04.1 LTS jorgserver tty1

jorgserver login: [ 83.232938] cloud-init[785]: Cloud-init v. 20.2-45-g5f7825e2-0ubuntu1~20.04.1 running 'modules:config' at Tue, 24 Nov 2020 14:41:41 +0000. Up 82.71 seconds.
[ 85.124490] cloud-init[792]: Cloud-init v. 20.2-45-g5f7825e2-0ubuntu1~20.04.1 running 'modules:final' at Tue, 24 Nov 2020 14:41:43 +0000. Up 84.15 seconds.
[ 85.124841] cloud-init[792]: Cloud-init v. 20.2-45-g5f7825e2-0ubuntu1~20.04.1 finished at Tue, 24 Nov 2020 14:41:44 +0000. Datasource DataSourceNone. Up 85.11 seconds
[ 85.125038] cloud-init[792]: 2020-11-24 14:41:44,043 - cc_final_message.py[WARNIN]: Used fallback datasource

jorgserver login: danibrogue
Password:
Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 5.4.0-48-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

* Introducing self-healing high availability clustering for MicroK8s!
  Super simple, hardened and opinionated Kubernetes for production.

      https://microk8s.io/high-availability

52 updates can be installed immediately.
0 of these updates are security updates.
To see these additional updates run: apt list --upgradable

The list of available updates is more than a week old.
To check for new updates run: sudo apt update

Last login: Sun Nov 22 15:39:17 UTC 2020 on tty1
danibrogue@jorgserver:~$ _
```

Рисунок 2 – Терминал после входа в систему.

3.2 Общая информация о системе

Для вывода информации о текущем интерпретаторе команд, необходимо написать “echo \$SHELL”

```
root@jorgserver:~# echo $SHELL  
/bin/bash  
root@jorgserver:~# _
```

Рисунок 3 – Информация о текущем интерпретаторе

Переменная окружения “SHELL” хранит путь до исполняемого файла оболочки.

Для вывода информации о текущем пользователе, необходимо написать команду “whoami”. Данная команда отображает имя пользователя, который вошел в систему

```
danibrogue@jorgserver:~$ whoami  
danibrogue  
danibrogue@jorgserver:~$ _
```

Рисунок 4 – Получение информации о текущем пользователе

Для получения информации о текущем каталоге используется команда “pwd”. Она выводит полный путь до текущей рабочей директории.

```
danibrogue@jorgserver:~$ pwd  
/home/danibrogue  
danibrogue@jorgserver:~$ _
```

Рисунок 5 – Получение информации о текущем каталоге

Чтобы посмотреть информацию об оперативной памяти и файле подкачки, необходимо ввести команду “free”. При её запуске без опций, она отобразит статистику в килобайтах.

```
danirogue@jorgserver:~$ free
total        used         free      shared  buff/cache   available
Mem:    1004848      152708      561548      1056      290592      701144
Swap:  1751036          0     1751036
danirogue@jorgserver:~$ _
```

Рисунок 6 – Информация об оперативной памяти и области подкачки
Где «Mem» – физическая память, «Swap» виртуальная память (Paging).

total — общее количество памяти;

used — реально использующаяся в данный момент и зарезервированная системой память;

free — свободная память ;

shared — Shared memory или Разделяемая память.

buffers — буферы в памяти — страницы памяти, зарезервированные системой для выделения их процессам, когда они затребуют этого.

cached — файлы, которые недавно были использованы системой/процессами и хранящиеся в памяти на случай если вскоре они снова потребуются.

С помощью команды “df -h” можно получить информацию о дисковой памяти. При применении опции “-h” данные будут выведены в более читаемом формате – мегабайтах и гигабайтах.

```
danirogue@jorgserver:~$ df -h
Filesystem           Size  Used Avail Use% Mounted on
udev                  449M    0  449M   0% /dev
tmpfs                 99M  1.1M  98M   2% /run
/dev/mapper/ubuntu--vg-ubuntu--lv  8.8G  4.3G  4.1G  52% /
tmpfs                 491M    0  491M   0% /dev/shm
tmpfs                 5.0M    0  5.0M   0% /run/lock
tmpfs                 491M    0  491M   0% /sys/fs/cgroup
/dev/loop1                56M    56M    0 100% /snap/core18/1885
/dev/loop0                56M    56M    0 100% /snap/core18/1932
/dev/sda2                976M  103M  806M  12% /boot
/dev/loop2                71M    71M    0 100% /snap/1xd/16922
/dev/loop3                68M    68M    0 100% /snap/1xd/18150
/dev/loop5                31M    31M    0 100% /snap/snapd/9721
/dev/loop4                31M    31M    0 100% /snap/snapd/9607
tmpfs                  99M    0  99M   0% /run/user/1000
danirogue@jorgserver:~$
```

Рисунок 7 – Информация о дисковой памяти

Где:

Filesystem – файловая система.

Size – размер в мегабайтах, показывается вся емкость точки монтирования.

Used – количество используемого дискового пространства.

Available – количество свободного пространства в мегабайтах.

Use% – процент использования файловой системы.

Mounted on – точка монтирования, где установлена файловая система.

3.3 Команды для получения информации о процессах

Для получения идентификатора текущего процесса (PID), необходимо использовать команду “echo \$\$”, где “\$\$”- идентификатор используемого в данный момент процесса командной оболочки.

```
danibrogue@jorgserver:~$ echo $$  
965  
danibrogue@jorgserver:~$ _
```

Рисунок 8 – Получение PID текущего процесса

Чтобы получить идентификатор родительского процесса (PPID), необходимо использовать команду “echo \$PPID”, где “\$PPID” – идентификатор соответствующего родительского процесса.

```
danibrogue@jorgserver:~$ echo $PPID  
648  
danibrogue@jorgserver:~$ _
```

Рисунок 9 – Получение PPID текущего процесса

Для получения идентификатора процесса инициализации системы, напишем команду “pidof init”. “init” – система инициализации в Unix – подобных системах, которая запускает все остальные процессы. Обычно, первый пользовательский процесс работает как демон и имеет идентификатор процесса – 1.

```
danibrogue@jorgserver:~$ pidof init  
1  
danibrogue@jorgserver:~$ _
```

Рисунок 10 – Получение PID процесса инициализации системы

Для получения информации о выполняющихся процессах текущего пользователя в текущем интерпретаторе команд, необходимо ввести команду “ps T -fu имя_пользователя”, где опция “T”- указывает на показ только тех процессов, которые связаны с текущим терминалом, опция “-f” выводит максимум доступных данных, а опция “-u” необходима для ограничения списка только процессами, запущенными определенным пользователем.

```
danibrogue@jorgserver:~$ ps T -fu danibrogue
UID      PID  PPID  C STIME TTY      STAT   TIME CMD
root     648      1  0 14:48 ttys000  Ss      0:00 /bin/login -p --
danibrogue  957      1  0 14:50 ?        Ss      0:00 /lib/systemd/systemd --user
danibrogue  959      957  0 14:50 ?        S      0:00 (sd-pam)
danibrogue  965      648  0 14:50 ttys000  S      0:00 -bash
danibrogue 1112      965  0 15:00 ttys000  R+     0:00 ps T -fu danibrogue
danibrogue@jorgserver:~$ _
```

Рисунок 11 – Процессы текущего пользователя и интерпретатора команд

Для отображения всех процессов, используем команду “ps -e |less”, где опция “-e” нужна, для просмотра всех запущенных процессов, а “|less” добавляет постраничный просмотр информации.

```
PID TTY      TIME CMD
 1 ?      00:00:03 systemd
 2 ?      00:00:00 kthreadd
 3 ?      00:00:00 rcu_gp
 4 ?      00:00:00 rcu_par_gp
 6 ?      00:00:00 kworker/0:0H-kblockd
 7 ?      00:00:01 kworker/0:1-events
 9 ?      00:00:00 mm_percpu_wq
10 ?      00:00:00 ksoftirqd/0
11 ?      00:00:01 rcu_sched
12 ?      00:00:00 migration/0
13 ?      00:00:00 idle_inject/0
14 ?      00:00:00 cpuhp/0
15 ?      00:00:00 kdevtmpfs
16 ?      00:00:00 netns
17 ?      00:00:00 rcu_tasks_kthre
18 ?      00:00:00 kauditd
19 ?      00:00:00 khungtaskd
20 ?      00:00:00 oom_reaper
21 ?      00:00:00 writeback
22 ?      00:00:00 kcompactd0
23 ?      00:00:00 ksmd
24 ?      00:00:00 khugepaged
70 ?      00:00:00 kintegrityd
71 ?      00:00:00 kblockd
72 ?      00:00:00 blkcg_punt_bio
73 ?      00:00:00 tpm_dev_wq
74 ?      00:00:00 ata_sff
75 ?      00:00:00 md
76 ?      00:00:00 edac-poller
77 ?      00:00:00 devfreq_wq
78 ?      00:00:00 watchdogd
79 ?      00:00:00 kworker/u2:1-events_power_efficient
81 ?      00:00:00 kswapd0
82 ?      00:00:00 ecryptfs-kthrea
84 ?      00:00:00 kthrotld
:-
```

Рисунок 12 – Просмотр всех процессов

Для выхода из постраничного просмотра, нажать клавишу “q”.

3.4 Команды управления процессами

Определить текущее значение nice по умолчанию возможно с помощью команды “nice”.

```
danibrogue@jorgserver:~$ nice  
0  
danibrogue@jorgserver:~$ _
```

Рисунок 13 – Просмотр значения nice

Во время создания процесса, каждой задаче присваивается статический приоритет, так же называемым правильным значением (nice value). При обычном запуске команд или программ, принимается равным приоритету родительского процесса. Значение nice может находиться в диапазоне от -20 до 19, где -20 – наивысший приоритет.

Для запуска интерпретатора bash с понижением приоритета, необходимо написать команду “nice -n 10 bash”. После чего запуститься интерпретатор с заданным приоритетом.

```
danibrogue@jorgserver:~$ ps -1  
F S  UID      PID  PPID C PRI  NI ADDR SZ WCHAN TTY          TIME CMD  
4 S 1000      965    648  0 80   0 - 2066 do_wai  tty1      00:00:00 bash  
0 R 1000     1170    965  0 80   0 - 2199 -          tty1      00:00:00 ps  
danibrogue@jorgserver:~$ nice -n 10 bash  
danibrogue@jorgserver:~$ ps -1  
F S  UID      PID  PPID C PRI  NI ADDR SZ WCHAN TTY          TIME CMD  
4 S 1000      965    648  0 80   0 - 2066 do_wai  tty1      00:00:00 bash  
0 S 1000     1173    965  1 90   10 - 2068 do_wai  tty1      00:00:00 bash  
0 R 1000     1180    1173  0 90   10 - 2199 -          tty1      00:00:00 ps  
danibrogue@jorgserver:~$ _
```

Рисунок 14 – Запуск bash с понижением приоритета

Для определения PID запущенного интерпретатора, необходимо использовать команду “pidof bash”.

```
danibrogue@jorgserver:~$ pidof bash  
1173 965  
danibrogue@jorgserver:~$ _
```

Рисунок 15 – Определение PID запущенного интерпретатора

Так же возможно определить PID командой “ps -f”.

```
danibrogue@jorgserver:~$ pidof bash  
1173 965  
danibrogue@jorgserver:~$ ps -f  
UID      PID  PPID   C STIME TTY          TIME CMD  
danibrogue+  965     648   0 14:50 ttys000    00:00:00 -bash  
danibrogue+  1173    965   0 15:04 ttys000    00:00:00 bash  
danibrogue+  1205    1173   0 15:05 ttys000    00:00:00 ps -f  
danibrogue@jorgserver:~$
```

Рисунок 16 – Определение PID по команде “ps -f”

Как можно увидеть, запущено 2 интерпретатора, поэтому результат получаем двумя PID.

Для замены текущего приоритета интерпретатора на приоритет, равным 5, необходимо использовать команду “renice -n 5 PID_процесса”. Данная команда позволяет изменять приоритет выполняемого процесса.

```
danibrogue@jorgserver:~$ ps -f  
UID      PID  PPID   C STIME TTY          TIME CMD  
danibrogue+  965     648   0 14:50 ttys000    00:00:00 -bash  
danibrogue+  1173    965   0 15:04 ttys000    00:00:00 bash  
danibrogue+  1234    1173   0 15:07 ttys000    00:00:00 ps -f  
danibrogue@jorgserver:~$ renice -n 5 1173  
renice: failed to set priority for 1173 (process ID): Permission denied  
danibrogue@jorgserver:~$ sudo renice -n 5 1173  
1173 (process ID) old priority 10, new priority 5  
danibrogue@jorgserver:~$ _
```

Рисунок 17 – Изменение приоритета

Как видно из рисунка, без root прав невозможно изменить приоритет в большую сторону, поэтому используем команду “**sudo renice -n 5 1173**”.

Для получения информации о процессах bash, необходимо использовать команду “ps lax |grep bash”.

```
danibrogue@jorgserver:~$ ps lax |grep bash
4 1000      965      648  20   0    8264  5260 do_wai S    tty1      0:00 -bash
0 1000     1173      965  25   5    8272  5076 do_wai SN   tty1      0:00 bash
0 1000     1265     1173  25   5    6300   664 -    RN+  tty1      0:00 grep --color=auto bash
danibrogue@jorgserver:~$
```

Рисунок 18 – Получение информации о процессах bash

4. Вывод

Выполнив лабораторную работу, на практике ознакомились со средствами управления процессами ОС Ubuntu.

5. Контрольные вопросы

1) Перечислите состояния задачи в ОС Ubuntu

Running (выполнение) – переходит в это состояние после выделения ей процессора.

Sleeping (спячка) – переходит в данное состояние после блокировки задачи.

Stopped (останов) – переходит в это состояние после остановки работы.

Zombie (зомби) – данное состояние показывает, что выполнение задачи прекратилось, но ещё не удалена из системы.

Dead (смерть) – Задача в этом состоянии, может быть удалена из системы

Active (активный) и expired (неактивный) используются при планировании выполнения процесса, поэтому они не сохраняются в переменной **state**.

2) Как создаются задачи в ОС Ubuntu?

Задачи создаются путем вызова системной функции `clone`. Любые обращения к `fork` или `vfork` преобразуются в системные вызовы `clone` во время компиляции. Функция `fork` создает дочернюю задачу, виртуальная память для которой выделяется по принципу копирования при записи (`copy-on-write`). Когда дочерний или же родительский процесс пытается выполнить запись в страницу памяти, записывающая программа создает собственную копию страницы в памяти.

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

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.

Цель алгоритма планирования состоит в том, чтобы обеспечить грубое пропорциональное соответствие качества обслуживания приоритету, то есть чем выше приоритет, тем меньше должно быть время отклика и тем большая доля процессорного времени достанется процессу.

5) Как можно изменить приоритет для выполняющейся задачи

Команда renice служит для изменения значения nice для уже выполняющихся процессов. Ее формат таков:

“renice priority [[-p]PID] [[-g] grp] [[-u] user]”