

Липецкий государственный технический университет

Факультет автоматизации и информатики

Кафедра Автоматизированных систем управления

Отчет по лабораторной работе № 4

«Управление процессами ОС Ubuntu»

по курсу «ОС Linux»

Студент
Группа АИ-18

Грунау Г.Ю.

Руководитель

Кургасов В.В.

Липецк 2020 г.

Цель работы

Ознакомиться со средствами управления процессами в Ubuntu ОС.

Содержание

Ход работы	4
1. Запуск Oracle VM VirtualBox	4
2. Запуск Ubuntu.....	5
3. Окно интерпретатора команд	5
4. Общая информация о системе.....	6
• Информация о текущем интерпретаторе	6
• Вывод информации о текущем пользователе.....	6
• Информация о текущем каталоге	6
• Информация об оперативной памяти и области подкачки	6
• Информация о дисковой памяти	7
5. Команды получения информации о процессах	9
• Идентификатор процесса оболочки	9
• Идентификатор родительского процесса оболочки	9
• Идентификатор процесса инициализации системы	9
• Информация о выполняющихся процессах текущего пользователя	10
• Вывод всех процессов.....	11
6. Команды управления процессами.....	12
• Определение текущего значения <code>ps</code> по умолчанию.	12
• Запуск <code>bash</code> с приоритетом 10	12
• Определение PID текущего интерпретатора	12
• Установка приоритета 5 запущенному интерпретатору	13
• Получение информации о процессах <code>bash</code>	13

Ход работы

1. Запуск Oracle VM VirtualBox

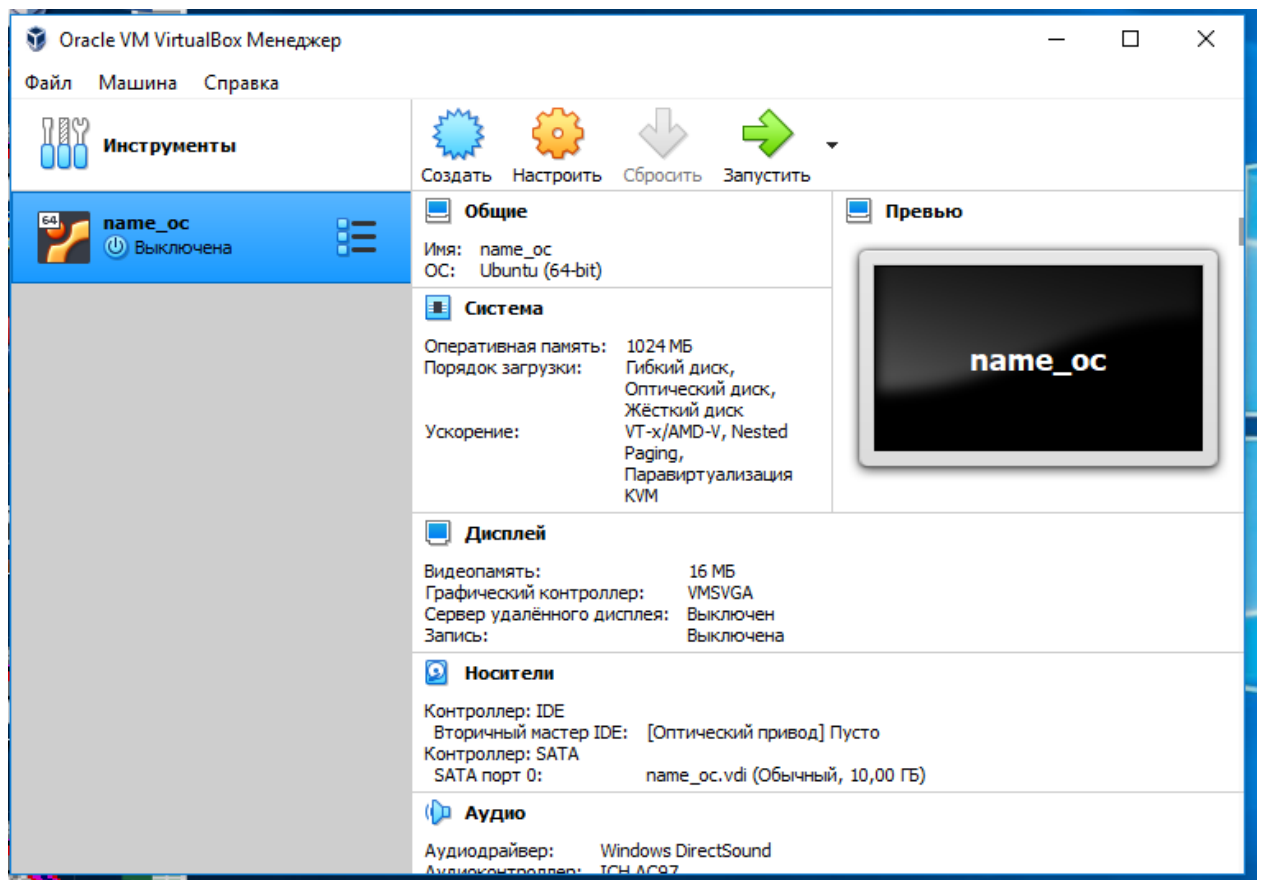


Рисунок 1 – Запуск Oracle VM VirtualBox

На Рисунок 1 изображён интерфейс программы Oracle VM VirtualBox. С её помощью я создал виртуальную машину name_os с операционной системой Ubuntu. На рисунке ниже изображен этап авторизации в терминале уже запущенной машины.

2. Запуск Ubuntu

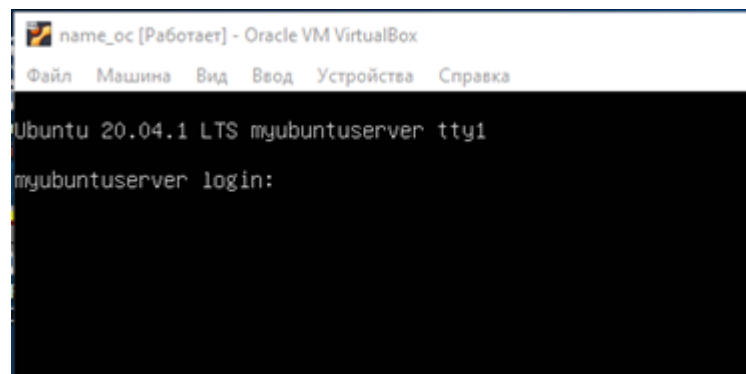


Рисунок 2 – Авторизация после запуска

3. Окно интерпретатора команд

После авторизации нам предоставляется окно интерпретатора команд:

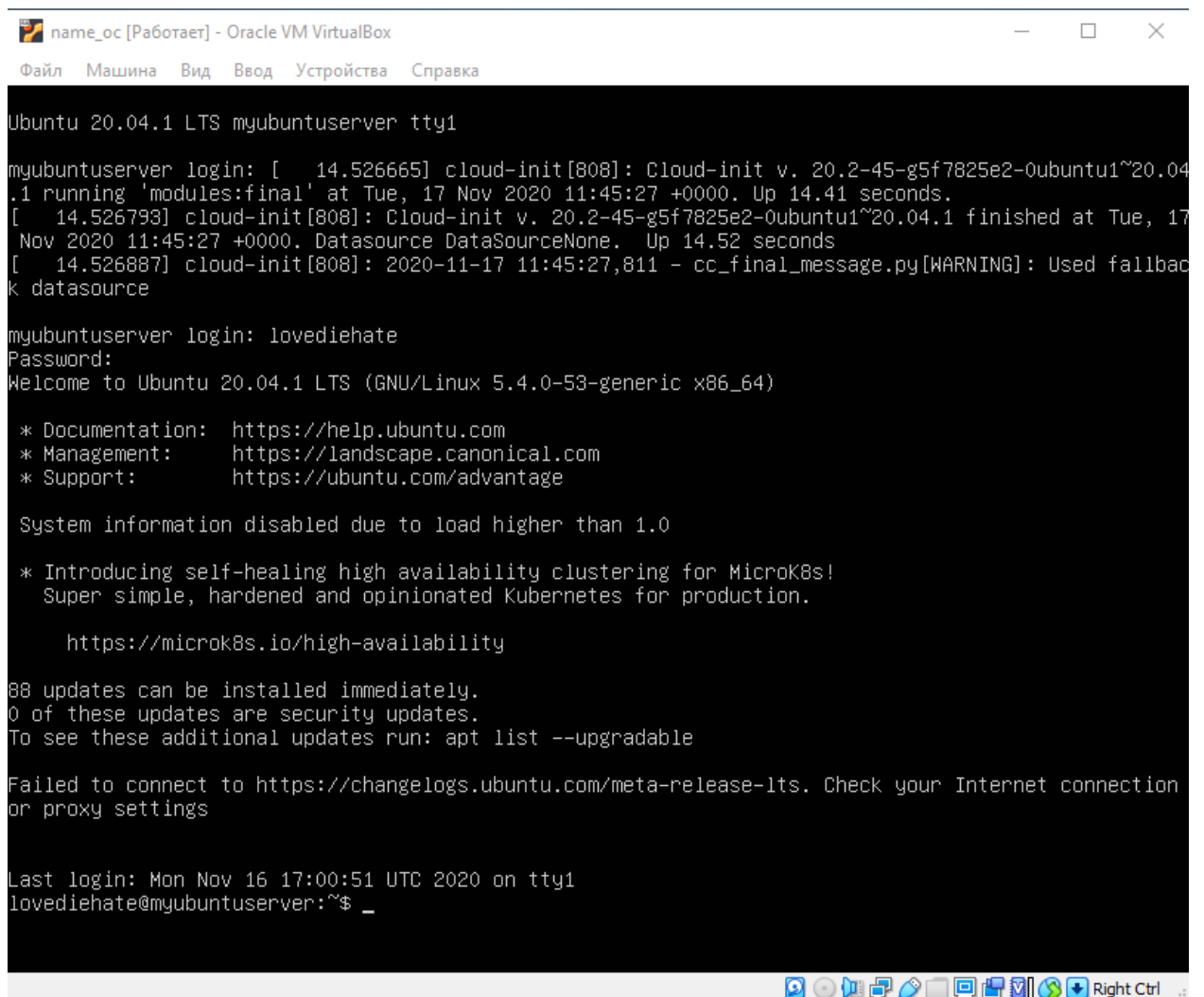


Рисунок 3 – Окно интерпретатора команд

4. Общая информация о системе

- Информация о текущем интерпретаторе

```
lovediehate@myubuntuserver:~$ echo $SHELL  
/bin/bash
```

Рисунок 4 – Информация об оболочке

На рисунке **Error! Reference source not found.** показано получение информации о текущем интерпретаторе с помощью команды `echo $SHELL`. Переменная окружения SHELL хранит путь до исполняемого файла оболочки. Из вывода команды, мы видим, что используется оболочка bash.

- Вывод информации о текущем пользователе

С помощью команды `whoami`, пользователь может узнать ответ на вопрос «Who am I?». В моем случае ответ lovediehate, т.к. я авторизовался с данной учётной записи.

```
lovediehate@myubuntuserver:~$ whoami  
lovediehate
```

Рисунок 5 – Команда whoami

- Информация о текущем каталоге

Команда `pwd` выводит путь к каталогу, в котором сейчас находится пользователь.

```
lovediehate@myubuntuserver:~$ pwd  
/home/lovediehate
```

Рисунок 6 – Команда pwd

- Информация об оперативной памяти и области подкачки

Вывод содержит данные о физической памяти Mem и файле подкачки Swap. В операционной системе Linux, как и в других ОС, файл подкачки нужен для страховки оперативной памяти. Когда установленный объем ОЗУ заканчивается, используется именно выделенная область из файла подкачки.

```
lovediehate@myubuntuserver:/home$ free
              total        used        free      shared  buff/cache   available
Mem:        1004848        157020        501324         1028       346504       694228
Swap:        1751036           0        1751036
```

Рисунок 7 – Команда free

В столбцах указаны следующие параметры:

Total – эта цифра представляет всю существующую память.

Used – вычисление общего значения оперативной памяти системы за вычетом выделенной свободной, разделяемой, буферной и кэш-памяти.

Free – это память, которая не используется ни для каких целей.

Shared, Buffer, и Cache – идентифицируют память, используемую для нужд ядра или операционной системы. Буфер и кэш складываются вместе, а сумма указывается в разделе «buff/cache».

Available – память появляется в более новых версиях free и предназначена для того, чтобы дать конечному пользователю оценку того, сколько ресурсов памяти все еще открыто для использования.

- Информация о дисковой памяти

Команда **df** предоставляет пользователю информацию о дисковой памяти. Параметр **-h** означает, что данные будут в мегабайтах и гигабайтах.

```
lovediehate@myubuntuserver:/home$ df -h
Filesystem      Size  Used Avail Use% Mounted on
udev            448M   0    448M   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/sda2       976M  197M   712M  22% /boot
/dev/loop0      56M   56M     0 100% /snap/core18/1932
/dev/loop2      68M   68M     0 100% /snap/lxd/18150
/dev/loop3      55M   55M     0 100% /snap/core18/1880
/dev/loop1      72M   72M     0 100% /snap/lxd/16099
/dev/loop4      30M   30M     0 100% /snap/snapd/8542
/dev/loop5      31M   31M     0 100% /snap/snapd/9721
tmpfs           99M   0    99M   0% /run/user/1000
```

Рисунок 8 – Команда df -h

Filesystem – файловая система.

Size – размер емкости точки монтирования в мегабайтах.

Used – количество используемого дискового пространства.

Available – количество свободного пространства в мегабайтах.

Use% – процент использования файловой системы.

Mounted on – точка монтирования, где установлена файловая система.

5. Команды получения информации о процессах

- Идентификатор процесса оболочки

```
lovediehate@myubuntuserver:/home$ echo $$
1039
lovediehate@myubuntuserver:/home$ ps -f
UID          PID     PPID  C  STIME TTY          TIME CMD
lovedie+      941       662  0  11:45 tty1        00:00:00 -bash
lovedie+     1039       941  0  11:50 tty1        00:00:00 /bin/bash
lovedie+     1636     1039  0  12:21 tty1        00:00:00 ps -f
```

Рисунок 9 – PID оболочки

Как видно на скриншоте, использовав команду `echo $$`, можно узнать PID текущей оболочки. В данном случае, оболочка `/bin/bash` имеет PID 1039.

- Идентификатор родительского процесса оболочки

Введя команду `echo $PPID`, можно узнать идентификатор родительского процесса оболочки:

```
lovediehate@myubuntuserver:/home$ ps -f
UID          PID     PPID  C  STIME TTY          TIME CMD
lovedie+      941       662  0  11:45 tty1        00:00:00 -bash
lovedie+     1039       941  0  11:50 tty1        00:00:00 /bin/bash
lovedie+     1636     1039  0  12:21 tty1        00:00:00 ps -f
lovediehate@myubuntuserver:/home$ echo $PPID
941
```

Рисунок 10 – PPID оболочки

- Идентификатор процесса инициализации системы

```
lovediehate@myubuntuserver:/home$ pidof init
1
```

Рисунок 11 – PID инициализации системы

Получаем идентификатора процесса по имени `init` с помощью команды `pidof init`. Init – система инициализации в Unix-подобных системах, которая запускает все остальные процессы. Первый пользовательский процесс работает как демон и обычно имеет PID 1.

- Информация о выполняющихся процессах текущего пользователя

```
lovediehate@myubuntuserver:/home$ ps T -fu lovediehate
UID      PID     PPID  C  STIME TTY      STAT   TIME CMD
root      662       1   0  11:45 tty1     Ss      0:00  /bin/login -p --
lovedie+  935       1   0  11:45 ?        Ss      0:00  /lib/systemd/systemd --user
lovedie+  936     935   0  11:45 ?        S       0:00  (sd-pam)
lovedie+  941     662   0  11:45 tty1     S       0:00  -bash
lovedie+ 1039     941   0  11:50 tty1     S       0:00  /bin/bash
lovedie+ 1885    1039   0  12:37 tty1     R+      0:00  ps T -fu lovediehate
```

Рисунок 12 – Процессы текущего пользователя

Команда `ps T -fu lovediehate` выводит информацию о процессах только текущего пользователя (параметр `-u lovediehate`) только в текущем интерпретаторе команд (параметр `T`).

- Вывод всех процессов

С помощью с параметра `-e` можно указать команде `ps` отобразить все процессы.

PID	TTY	TIME	CMD
1	?	00:00:01	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
9	?	00:00:00	mm_percpu_wq
10	?	00:00:00	ksoftirqd/0
11	?	00:00:00	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
81	?	00:00:00	kswapd0
82	?	00:00:00	ecryptfs-kthrea
84	?	00:00:00	kthrotld
85	?	00:00:00	acpi_thermal_pm
86	?	00:00:00	scsi_eh_0

Рисунок 13 – Вывод всех процессов

6. Команды управления процессами

- Определение текущего значения nice по умолчанию.

```
lovediegate@myubuntuserver:/home$ nice  
0
```

Рисунок 14 – Текущее значение nice

В нашем случае значение nice по умолчанию оказалось равным 0.

Во время создания каждой задаче присваивается статический приоритет (static priority), называемый также правильным значением (nice value). При обычном запуске команд или программ принимается равным приоритету родительского процесса.

Значение nice находится в диапазоне от -20 до 19. Большее значение означает меньший приоритет.

- Запуск bash с приоритетом 10

```
lovediegate@myubuntuserver:/home$ nice -n 10 bash  
lovediegate@myubuntuserver:/home$ ps -l
```

	F	S	UID	PID	PPID	C	PRI	NI	ADDR	SZ	WCHAN	TTY	TIME	CMD
4	S	1000	941	662	0	80	0	0	1768	do_wai	tty1	00:00:00	bash	
0	S	1000	1039	941	0	80	0	0	1760	do_wai	tty1	00:00:00	bash	
0	S	1000	2169	1039	0	90	10	0	1760	do_wai	tty1	00:00:00	bash	
0	S	1000	2180	2169	0	99	19	0	1760	do_wai	tty1	00:00:00	bash	
0	R	1000	2190	2180	0	99	19	0	1888	-	tty1	00:00:00	ps	

Рисунок 15 – Запуск bash с понижением приоритета

На рисунке 15 видно, что значение NI у процесса bash поменялось на 10.

- Определение PID текущего интерпретатора

```
lovediegate@myubuntuserver:/home$ pidof bash  
2180 2169 1039 941
```

Рисунок 16 – Команда pidof

На рисунке 16 видно, что запущено 4 интерпретатора bash.

- Установка приоритета 5 запущенному интерпретатору

```
lovediehatemyubuntuserver:/home$ ps -l && sudo renice -n 5 2180 && ps -l
F S  UID      PID     PPID  C PRI  NI ADDR SZ WCHAN  TTY          TIME CMD
4 S  1000      941      662   0  80   0  - 1768 do_wai  tty1        00:00:00 bash
0 S  1000     1039      941   0  80   0  - 1760 do_wai  tty1        00:00:00 bash
0 S  1000     2169     1039   0  90  10  - 1760 do_wai  tty1        00:00:00 bash
0 S  1000     2180     2169   0  99  19  - 1760 do_wai  tty1        00:00:00 bash
0 R  1000     2445     2180   0  99  19  - 1888 -        tty1        00:00:00 ps
2180 (process ID) old priority 19, new priority 5
F S  UID      PID     PPID  C PRI  NI ADDR SZ WCHAN  TTY          TIME CMD
4 S  1000      941      662   0  80   0  - 1768 do_wai  tty1        00:00:00 bash
0 S  1000     1039      941   0  80   0  - 1760 do_wai  tty1        00:00:00 bash
0 S  1000     2169     1039   0  90  10  - 1760 do_wai  tty1        00:00:00 bash
0 S  1000     2180     2169   0  85   5  - 1760 do_wai  tty1        00:00:00 bash
0 R  1000     2448     2180   0  85   5  - 1888 -        tty1        00:00:00 ps
```

Рисунок 17 – Установка приоритета

С помощью команды `sudo renice -n 5 2180` мы можем повысить приоритет процессу. Для понижения приоритета администраторские права не требуются.

- Получение информации о процессах bash

```
lovediehatemyubuntuserver:/home$ ps lax | grep bash
4 1000      941      662  20   0 7072 5108 do_wai S    tty1        0:00 -bash
0 1000     1039      941  20   0 7040 5064 do_wai S    tty1        0:00 /bin/bash
0 1000     2169     1039  30  10 7040 4920 do_wai SN   tty1        0:00 bash
0 1000     2180     2169  25   5 7040 5028 do_wai SN   tty1        0:00 bash
0 1000     2475     2180  25   5 5192  736 -      RN+  tty1        0:00 grep --color=auto bash
```

Рисунок 18 – Информация о процессах bash

С помощью утилиты `grep bash`, мы можем отобразить только нужные нам процессы, связанные с bash.