

Отчёт по лабораторной работе №1

Основы информационной безопасности

Чистов Даниил Максимович

Содержание

1	Цель работы	4
2	Задания	5
3	Выполнение лабораторной работы	6
3.1	Настройка виртуальной машины и установка операционной системы	6
3.2	Домашнее задание	16
4	Выводы	19
	Список литературы	20

Список иллюстраций

3.1	Первоначальные настройки виртуальной машины	6
3.2	Настройка виртуальной машины	7
3.3	Количество памяти	7
3.4	Финальные параметры виртуальной машины	8
3.5	Начало установки операционной системы	8
3.6	Язык операционной системы	9
3.7	Часовой пояс виртуальной машины	9
3.8	Server with GUI, Development tools	10
3.9	Настройка клавиатуры	10
3.10	Отключение KDUMP	11
3.11	Настройка сети	11
3.12	пароль для root-доступа	12
3.13	Новый пользователь	12
3.14	Жёсткий диск для операционной системы	13
3.15	Начальные настройки ОС	13
3.16	Установка	14
3.17	Установка завершена	14
3.18	Образа больше нет	15
3.19	Подтверждение лицензионного соглашения	15
3.20	Подключение дополнительных функций	16
3.21	Вывод команды dmesg less	17
3.22	Linux Version, Mhz processor, CPU0	17
3.23	кол-во доступной памяти, тип гипервизора, тип файловой системы, последовательность монтирования файловых систем	18

1 Цель работы

Целью данной работы является приобретение практических навыков установки операционной системы на виртуальную машину, настройки минимально необходимых для дальнейшей работы сервисов

2 Задания

1. Настройка виртуальной машины и установка операционной системы
2. Домашнее задание

3 Выполнение лабораторной работы

3.1 Настройка виртуальной машины и установка операционной системы

Я выполнял работу дома на стационарном компьютере, пользовался VirtualBox, по заданию требуется поставить ОС Rocky, я скачал её образ с официального сайта. Перейдём к процессу настройки виртуальной машины.

Задаю имя виртуальной машины согласно требованиям к наименованию, также задаю путь и выбираю нужный образ, скачанный с интернета (рис. 3.1).

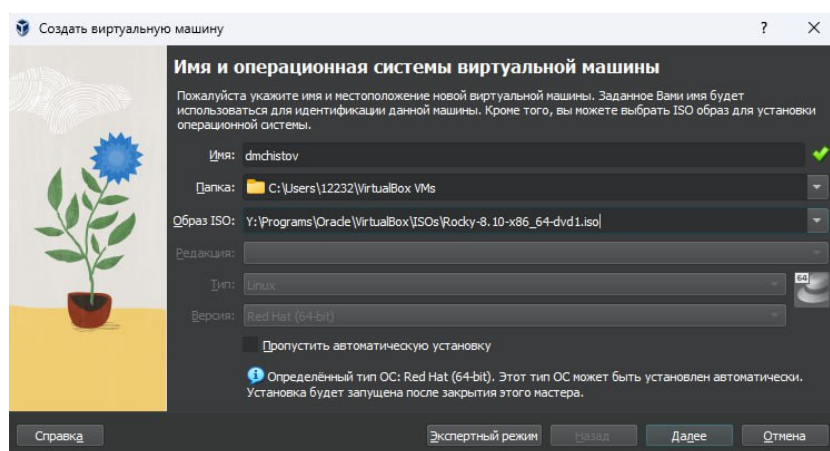


Рис. 3.1: Первоначальные настройки виртуальной машины

Выбираю кол-во процессоров и выделяю нужное мне кол-во памяти (рис. 3.2).

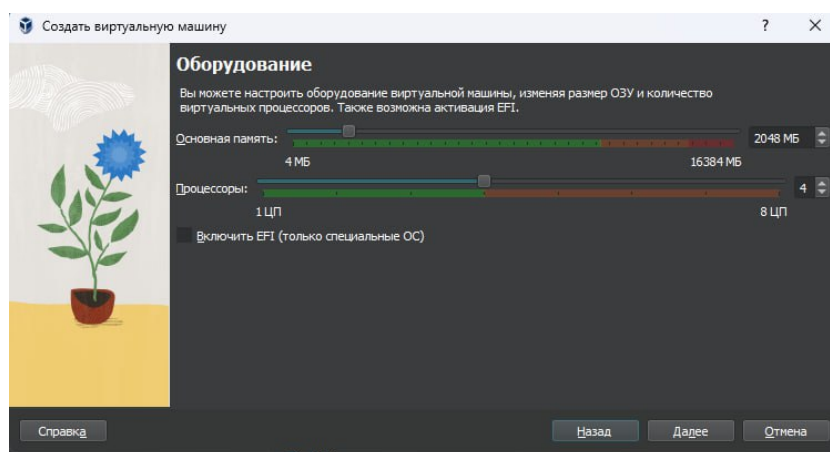


Рис. 3.2: Настройка виртуальной машины

Выделяю нужное количество памяти жёсткого диска для моей виртуальной машины (рис. 3.3).

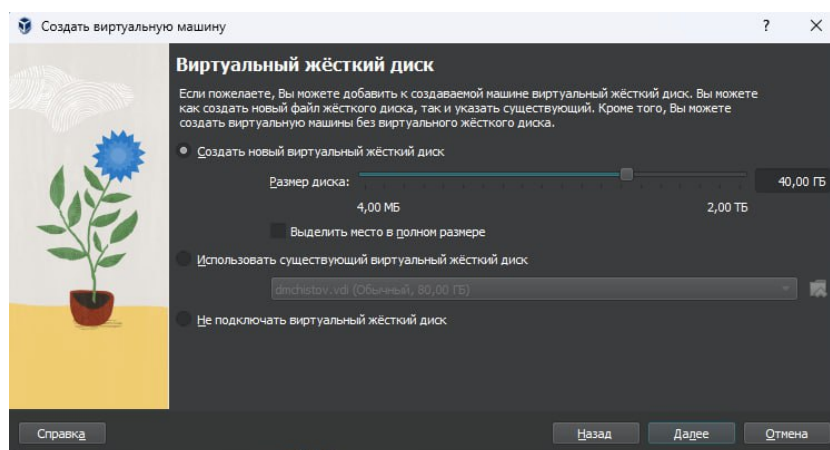


Рис. 3.3: Количество памяти

Итоговые настройки моей виртуальной машины (рис. 3.4).

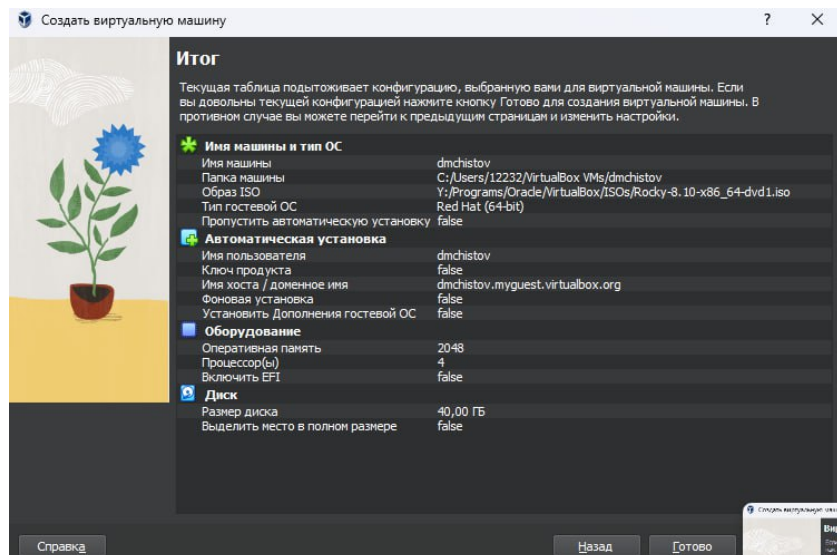


Рис. 3.4: Финальные параметры виртуальной машины

Запускаю виртуальную машину и вижу, что установка успешно начата (рис. 3.5).

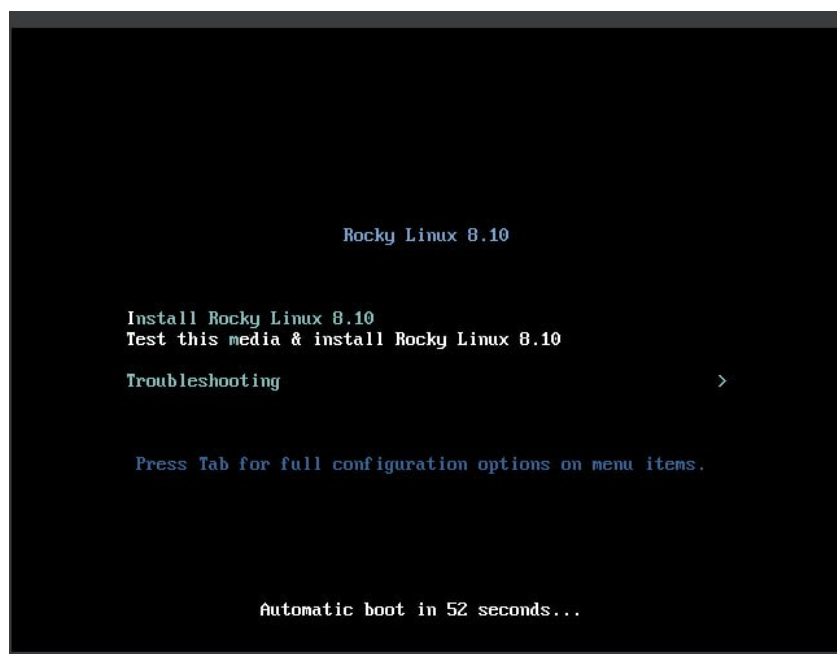


Рис. 3.5: Начало установки операционной системы

Задаю язык операционной системы (рис. 3.6).

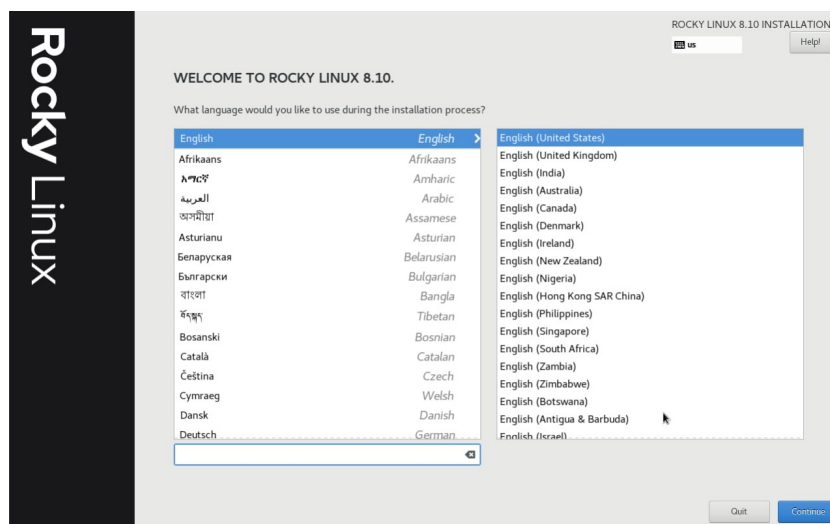


Рис. 3.6: Язык операционной системы

Задаю часовой пояс операционной системы - Москва, Московское время(рис. 3.7).

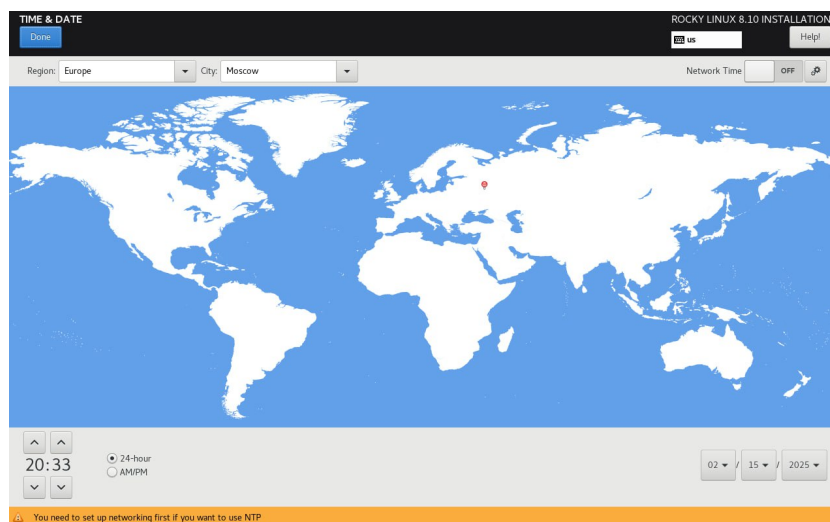


Рис. 3.7: Часовой пояс виртуальной машины

В разделе выбора программ, указываю базовое окружение - Server with GUI, а в качестве дополнения - Development tools (рис. 3.8).

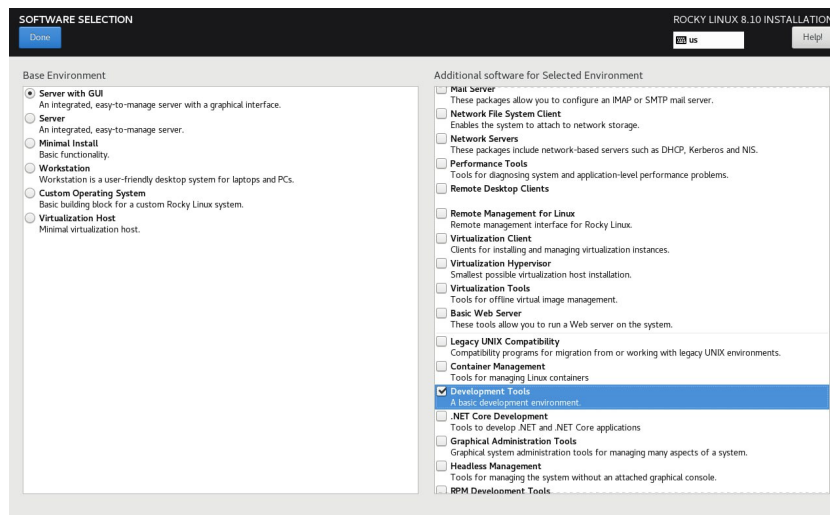


Рис. 3.8: Server with GUI, Development tools

Настроиваю клавиатуру - выбираю раскладку: русскую и английскую (рис. 3.9).

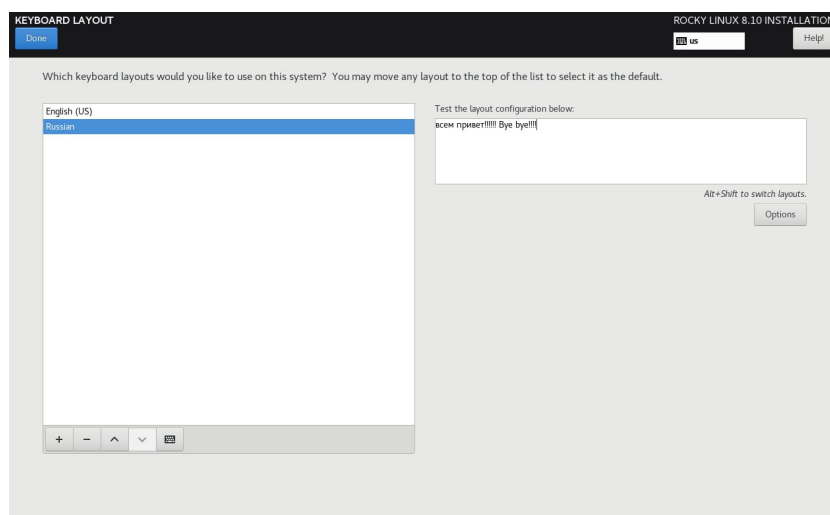


Рис. 3.9: Настройка клавиатуры

Отключаю KDUMP (рис. 3.10).

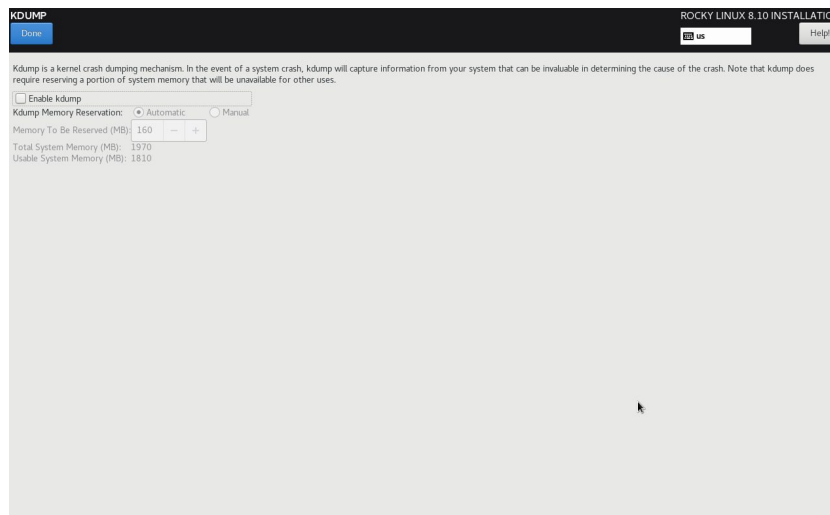


Рис. 3.10: Отключение KDUMP

Включаю сетевое соединение, а в качестве имени узла указываю dmchistov.localdomain (рис. 3.11).

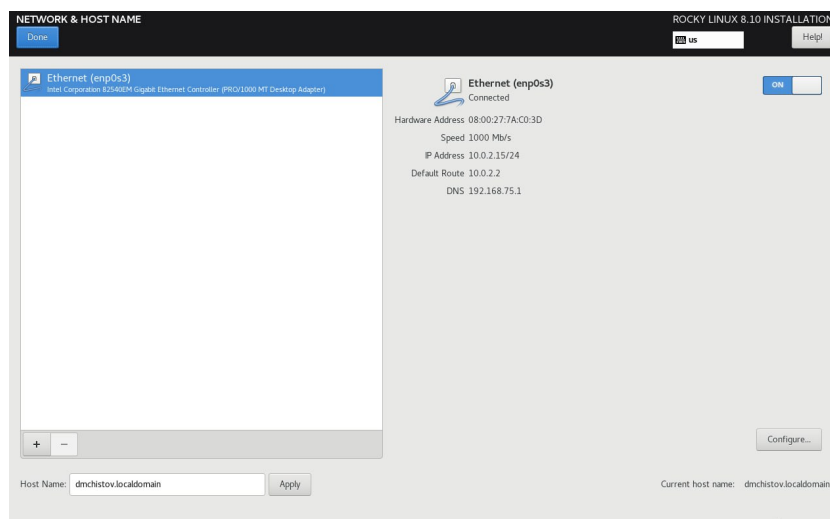


Рис. 3.11: Настройка сети

Устанавливаю пароль для root (рис. 3.12).

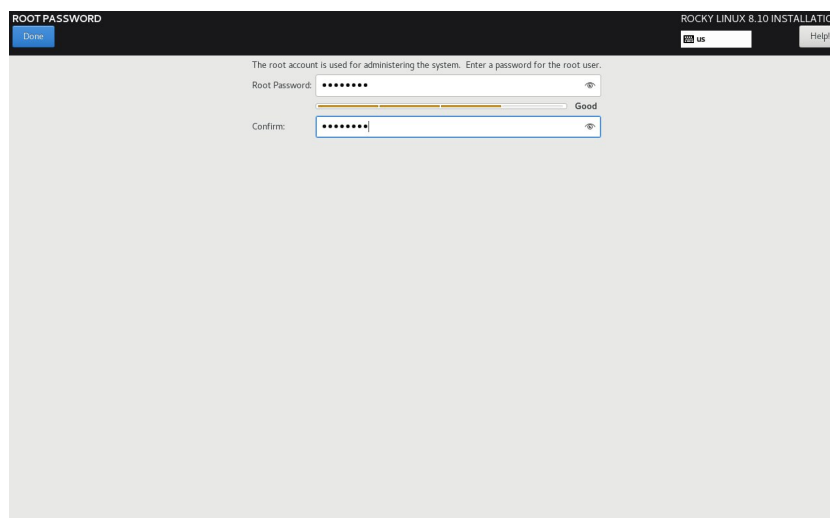


Рис. 3.12: пароль для root-доступа

Создаю пользователя dmchistov и задаю ему пароль (рис. 3.13).

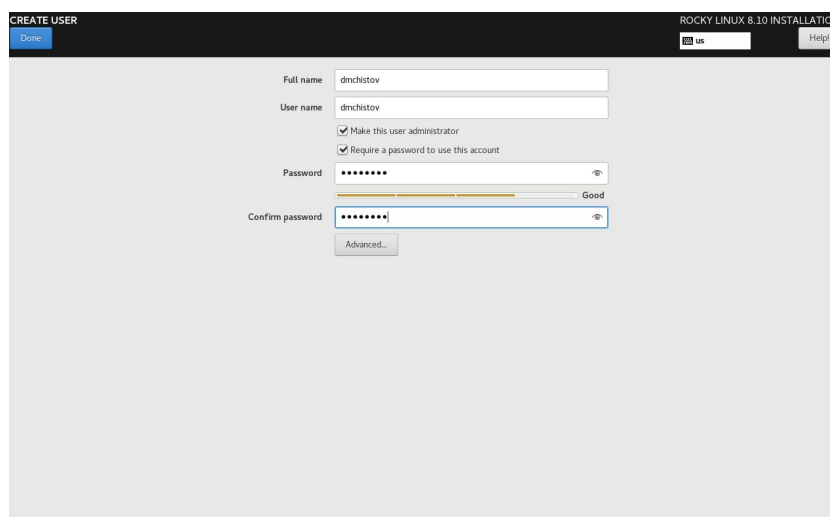


Рис. 3.13: Новый пользователь

Выбираю свой виртуальный жёсткий диск (рис. 3.14).

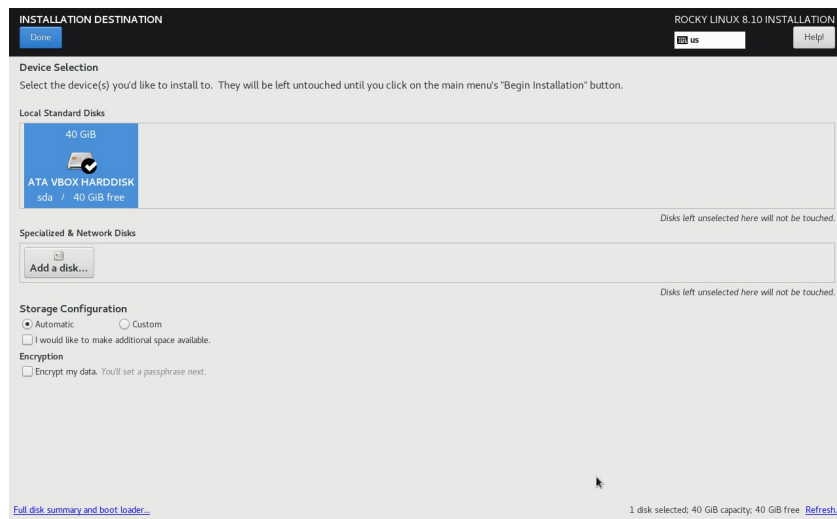


Рис. 3.14: Жёсткий диск для операционной системы

Начальная настройка операционной системы завершена, перейду к установке (рис. 3.15).



Рис. 3.15: Начальные настройки ОС

Установка (рис. 3.16).

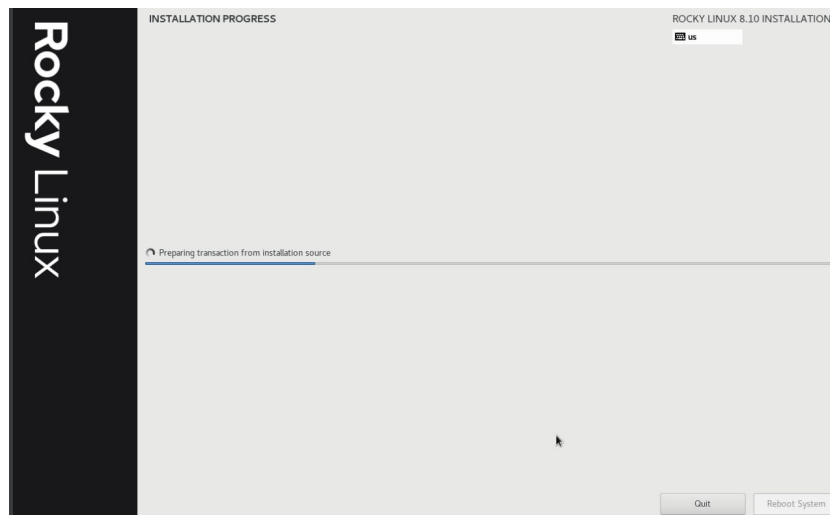


Рис. 3.16: Установка

Установка завершена (рис. 3.17).



Рис. 3.17: Установка завершена

Проверяю, что образ успешно автоматически исчез после установки и больше и не используется (рис. 3.18).

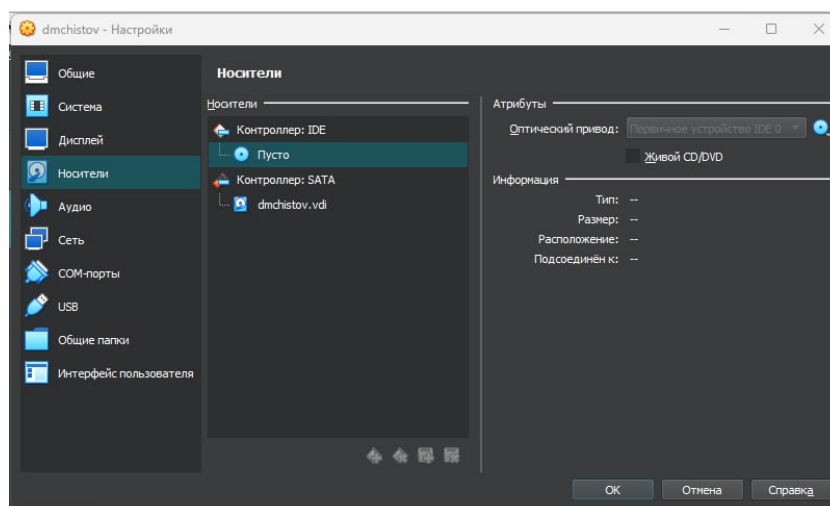


Рис. 3.18: Образа больше нет

Подтверждаю лицензионное соглашение (рис. 3.19).

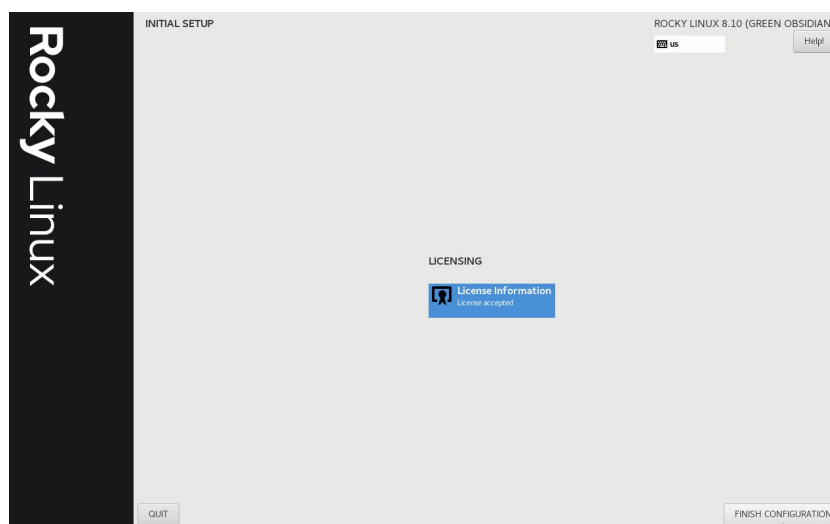


Рис. 3.19: Подтверждение лицензионного соглашения

После успешной установки, также подключу образ с дополнениями к моей ОС (рис. 3.20).

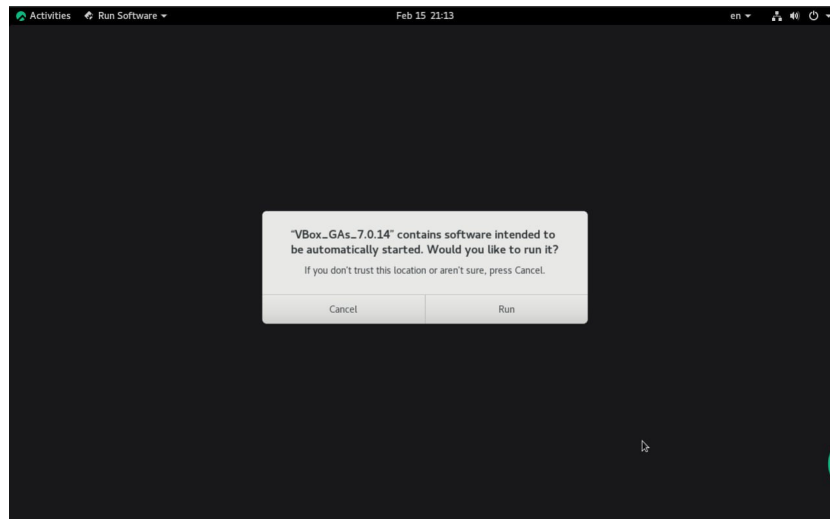


Рис. 3.20: Подключение дополнительных функций

3.2 Домашнее задание

По заданию мне требуется узнать утилитой `dmesg` следующие вещи:

1. Версия ядра Linux (Linux version).
2. Частота процессора (Detected Mhz processor).
3. Модель процессора (CPU0).
4. Объем доступной оперативной памяти (Memory available).
5. Тип обнаруженного гипервизора (Hypervisor detected).
6. Тип файловой системы корневого раздела.
7. Последовательность монтирования файловых систем.

По мере выполнения задания я использовал информацию в интернете о команде `dmesg`, ссыла будет прикреплена в источниках.

Пока просто посмотрю вывод этой команды, написав `dmesg | less` (рис. 3.21).


```
dmchistov@dmchistov:~$ dmesg | less
0.000000 Linux version 4.18.0-553.el8_10.x86_64 (mockbuild@iad1-prod-build001.bld.equ.rockylinux.org) (gcc version 8.5.
20210514 (Red Hat 8.5.0-22) (GCC)) #1 SMP Fri May 24 13:05:10 UTC 2024
0.000000 Command line: BOOT_IMAGE=(hdd,msdosl)/vmlinuz-4.18.0-553.el8_10.x86_64 root=/dev/mapper/rh_dmchistov-root ro r
sume=/dev/mapper/rh_dmchistov-swap rd.lvm.lv=rh_dmchistov/root rd.lvm.lv=rh_dmchistov/swap rhgb quiet
0.000000 x86/fpu: Supporting XSAVE feature 0x001: 'x87 floating point registers'
0.000000 x86/fpu: Supporting XSAVE feature 0x002: 'SSE registers'
0.000000 x86/fpu: Supporting XSAVE feature 0x004: 'AVX registers'
0.000000 x86/fpu: xstate offset[2]: 576, xstate sizes[2]: 256
0.000000 x86/fpu: Enabled xstate features 0x7, context size is 832 bytes, using 'standard' format.
0.000000 signal: max sigframe size: 1776
0.000000 BIOS-provided physical RAM map:
0.000000 BIOS-e820: [mem 0x0000000000000000-0x0000000000000bfff] usable
0.000000 BIOS-e820: [mem 0x0000000000000c00-0x0000000000000ffff] reserved
0.000000 BIOS-e820: [mem 0x0000000000000f0000-0x0000000000000ffff] reserved
0.000000 BIOS-e820: [mem 0x000000000000100000-0x00000000000007ffff] usable
0.000000 BIOS-e820: [mem 0x000000000000100000-0x00000000000007ffff] ACPI data
0.000000 BIOS-e820: [mem 0x0000000000000fec00000-0x0000000000000ffff] reserved
0.000000 BIOS-e820: [mem 0x0000000000000fec00000-0x0000000000000ffff] reserved
0.000000 BIOS-e820: [mem 0x0000000000000ffff0000-0x0000000000000ffff] reserved
0.000000 NX (Execute Disable) protection: active
0.000000 SMBIOS 2.5 present.
0.000000 DMI: Innovek GmbH VirtualBox/VirtualBox, BIOS VirtualBox 12/01/2006
0.000000 Hypervisor detected: KVM
0.000000 kvm-clock: Using msrc 4b564d01 and 4b564d00
0.000000 kvm-clock: using sctd offset of 6389478735 cycles
0.000000 clocksource: kvm-clock: mask: 0xffffffffffff max_cycles: 0x1cd42e4dffb, max_idle_ns: 881590591483 ns
0.000000 tsc: Detected 3600.008 MHz processor
0.000000 e820: update [mem 0x00000000-0x00000fff] usable ==> reserved
0.000000 e820: remove [mem 0x00000000-0x00000fff] usable
0.000000 last_pfn = 0x7ffff0 max_arch_pfn = 0x40000000
0.000000 MTRR default types: uncachable
0.000000 MTRR variable ranges disabled:
0.000000 Disabled
0.000000 x86/PAT: MTRRs disabled, skipping PAT initialization too.
0.000000 CPU MTRRs all blank - virtualized system.
0.000000 x86/PAT: Configuration [0-7]: WB WT UC- UC WB WT UC- UC
0.000000 found SMP MP-table at [mem 0x0009ffff-0x0009ffff]
0.000000 BRK [0x46801000, 0x46801fff] PGTABLE
0.000000 BRK [0x46802000, 0x46802fff] PGTABLE
```

Рис. 3.21: Вывод команды dmesg | less

Используя `dmesg | grep -i ""` (в кавычках пишу то, что ищу), нахожу всё, что мне надо по заданию. Для начала - Linux Version, Mhz processor, CPU0 (рис. 3.22).

```
dmchistov@dmchistov:~$ dmesg | grep -i "Linux version"
0.000000 Linux version 4.18.0-553.el8_10.x86_64 (mockbuild@iad1-prod-build001.bld.equ.rockylinux.org) (gcc vers
on 8.5.0 20210514 (Red Hat 8.5.0-22) (GCC)) #1 SMP Fri May 24 13:05:10 UTC 2024
dmchistov@dmchistov:~$ dmesg | grep -i "Detected Mhz processor"
0.000000 tsc: Detected 3600.008 MHz processor
dmchistov@dmchistov:~$ dmesg | grep -i "Detected Mhz"
dmchistov@dmchistov:~$ dmesg | grep -i "Detected"
0.000000 Hypervisor detected: KVM
0.000000 tsc: Detected 3600.008 MHz processor
0.838679 hub 1-0:1.0: 12 ports detected
0.891799 hub 2-0:1.0: 12 ports detected
1.158640 systemd[1]: Detected virtualization oracle.
1.158643 systemd[1]: Detected architecture x86_64.
3.138116 systemd[1]: Detected virtualization oracle.
3.138119 systemd[1]: Detected architecture x86_64.
dmchistov@dmchistov:~$ dmesg | grep -i "CPU0"
0.117054 smpboot: CPU0: Intel(R) Core(TM) i3-8100 CPU @ 3.60GHz (family: 0x6, model: 0x9e, stepping: 0xb)
```

Рис. 3.22: Linux Version, Mhz processor, CPU0

Также узнаю о кол-ве доступной памяти, типе гипервизора, типе файловой системы и о последовательности монтирования файловых систем (рис. 3.23).

```

0.000000 PM: Registered nosave memory: [mem 0x000a0000-0x000effff]
0.000000 PM: Registered nosave memory: [mem 0x000f0000-0x000fffff]
0.000000 Memory: 261120K/2096696K available (14339K kernel code, 5957K rwdats, 8568K rodata, 2820K init, 13792K b
s, 13844K reserved, 0K cma-reserved)
0.014966 Freeing SMP alternatives memory: 36K
0.128194 x86/mm: Memory block size: 128MB
0.740037 Freeing initrd memory: 50468K
0.829112 Non-volatile memory driver v1.3
1.144436 Freeing unused decrypted memory: 2028K
1.144823 Freeing unused kernel image (initmem) memory: 2820K
1.149638 Freeing unused kernel image (text/rodata gap) memory: 2016K
1.149936 Freeing unused kernel image (rodata/data gap) memory: 1672K
1.778461 vmwgfx 0000:00:02:0: [drm] Legacy memory limits: VRAM = 16384 kB, FIFO = 2048 kB, surface = 507904 kB
1.778466 vmwgfx 0000:00:02:0: [drm] Maximum display memory size is 16384 kiB
dmchistov@dmchistov ~]$ dmesg | grep -i "Hypervisor detected"
0.000000 Hypervisor detected: KVM
dmchistov@dmchistov ~]$ dmesg | grep -i "sda"
2.052076 sd 2:0:0:0: [sda] 83886080 512-byte logical blocks: (42.9 GB/40.0 GiB)
2.052084 sd 2:0:0:0: [sda] Write Protect is off
2.052085 sd 2:0:0:0: [sda] Mode Sense: 00 3a 00 00
2.052092 sd 2:0:0:0: [sda] Write cache: enabled, read cache: enabled, doesn't support DPO or FUA
2.053395 sda: sda1 sda2
2.053737 sd 2:0:0:0: [sda] Attached SCSI disk
4.569939 XFS (sda1): Mounting V5 Filesystem
4.870479 XFS (sda1): Ending clean mount
dmchistov@dmchistov ~]$ dmesg | grep -i "Mount"
0.004018 Mount-cache hash table entries: 4096 (order: 3, 32768 bytes, vmalloc)
0.004023 Mountpoint-cache hash table entries: 4096 (order: 3, 32768 bytes, vmalloc)
2.531326 XFS (dm-0): Mounting V5 Filesystem
2.548943 XFS (dm-0): Ending clean mount
4.569939 XFS (sda1): Mounting V5 Filesystem
4.870479 XFS (sda1): Ending clean mount
dmchistov@dmchistov ~]$

```

Рис. 3.23: кол-во доступной памяти, тип гипервизора, тип файловой системы, последовательность монтирования файловых систем

4 Выводы

При выполнении данной лабораторной работы я приобрёл практические навыки установки операционной системы на виртуальную машину, настройки минимально необходимых для дальнейшей работы сервисов

Список литературы

Лабораторная работы №1

Команда dmesg