**РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ**

**Факультет физико-математических и естественных наук**

**Кафедра прикладной информатики и теории вероятностей**

**ОТЧЕТ**

**ПО ЛАБОРАТОРНОЙ РАБОТЕ № 1**

*дисциплина: Операционные системы*

Студент:

Нитусова Диана Денисовна

Группа:

НПМбд-01-20

**МОСКВА**

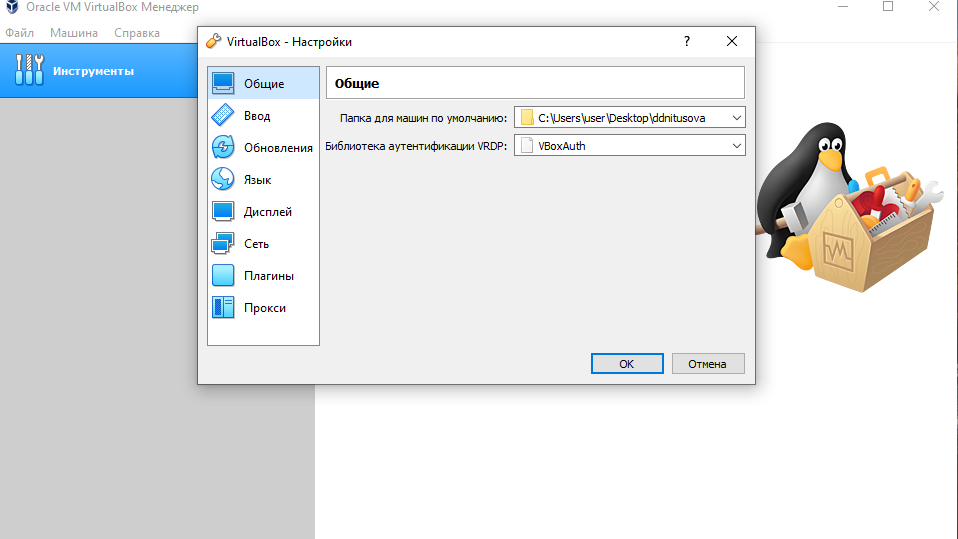
2020 г.

**1. Цель работы:**

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

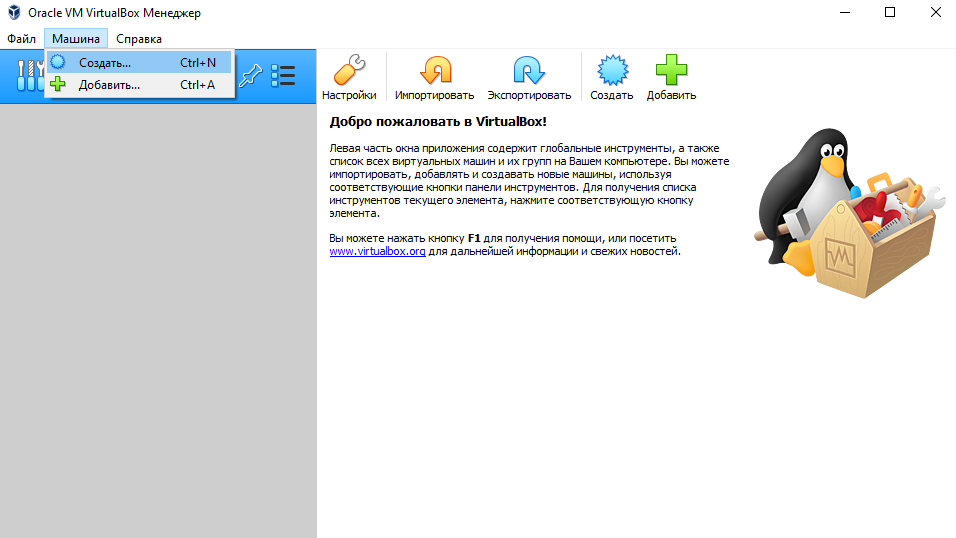
**2. Ход работы:**

Создаём на рабочем столе папку, в которой будет храниться наша виртуальная машина. Имя папки – ddnitusova. Проверяем в свойствах VirtualBox месторасположение папки для виртуальных машин. Для этого открываем VirtualBox, далее «Файл» → «Свойства» → вкладка «Общие» и в поле «Папка для машин по умолчанию» указываем путь к папке (Рисунок 1).

****

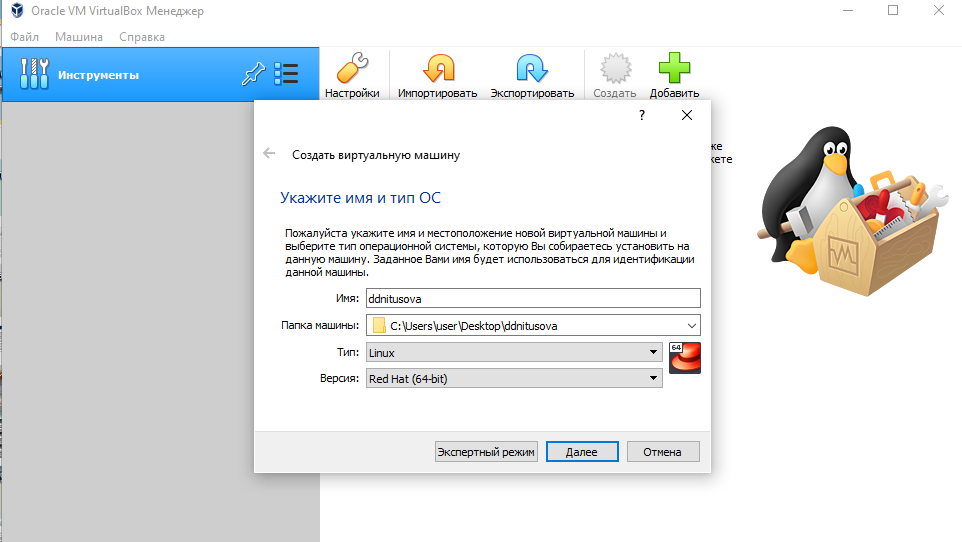
(Рисунок 1)

Переходим к созданию виртуальной машины. Для этого нажимаем «Машина» → «Создать» (Рисунок 2).



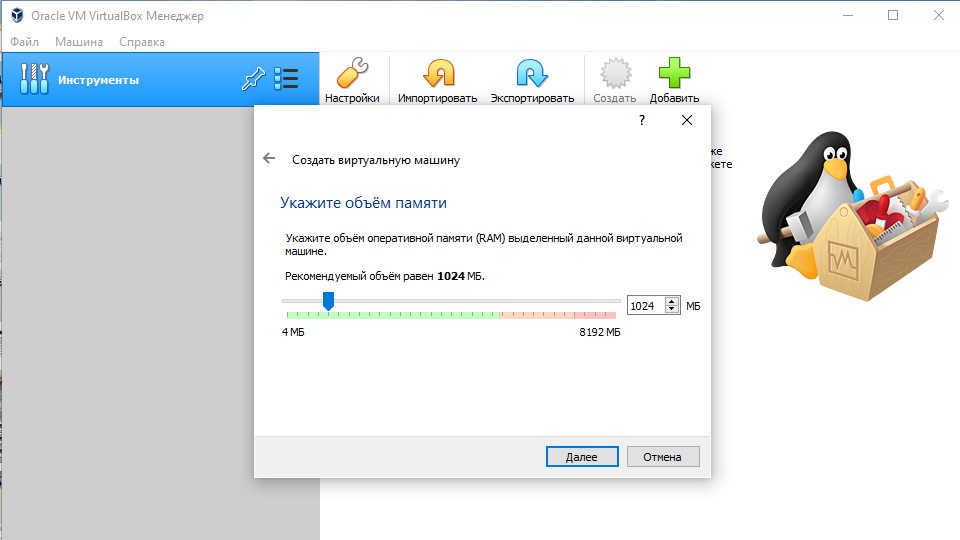
(Рисунок 2)

Указываем имя виртуальной машины и тип операционной системы – Linux, RedHat (64-bit) (Рисунок 3).



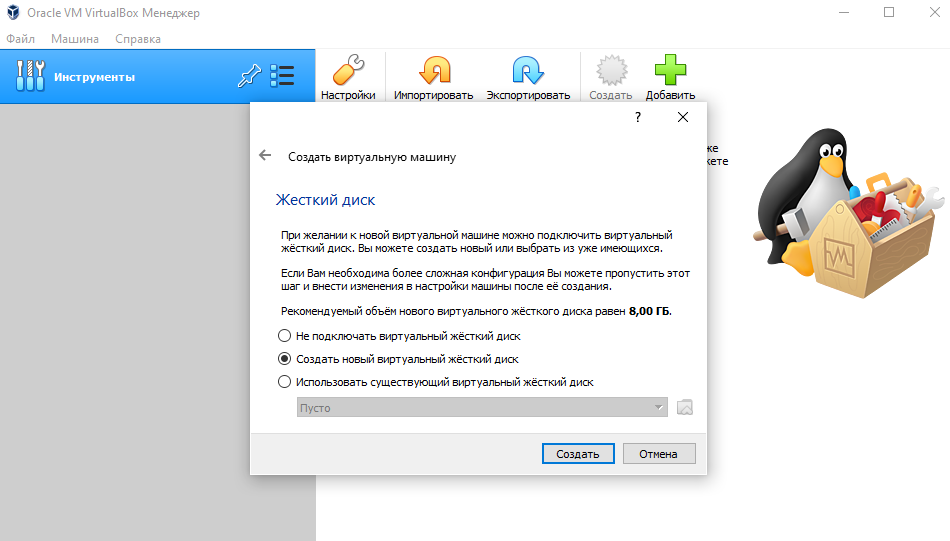
(Рисунок 3)

Указываем размер основной памяти виртуальной машины – 1024 МБ (Рисунок 4).



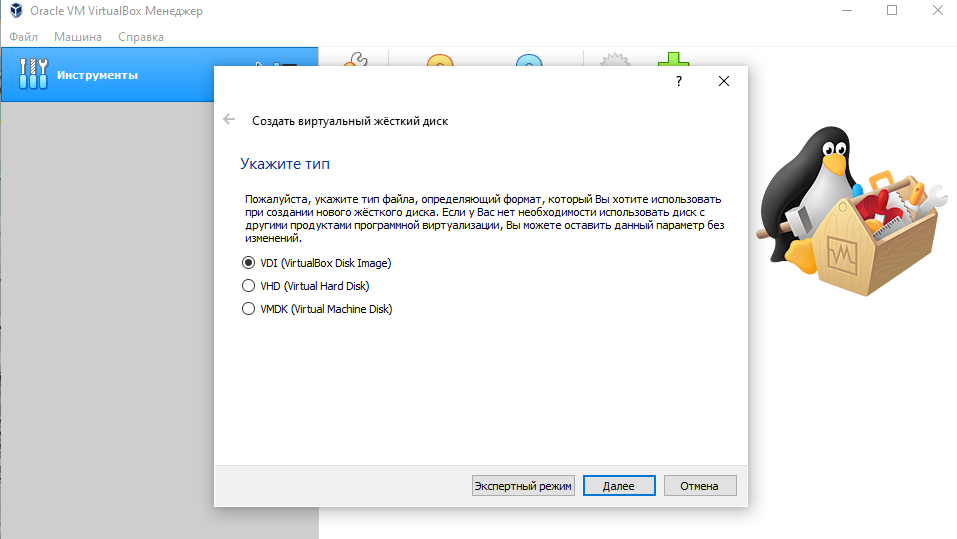
(Рисунок 4)

Создаём новый виртуальный жёсткий диск (Рисунок 5).

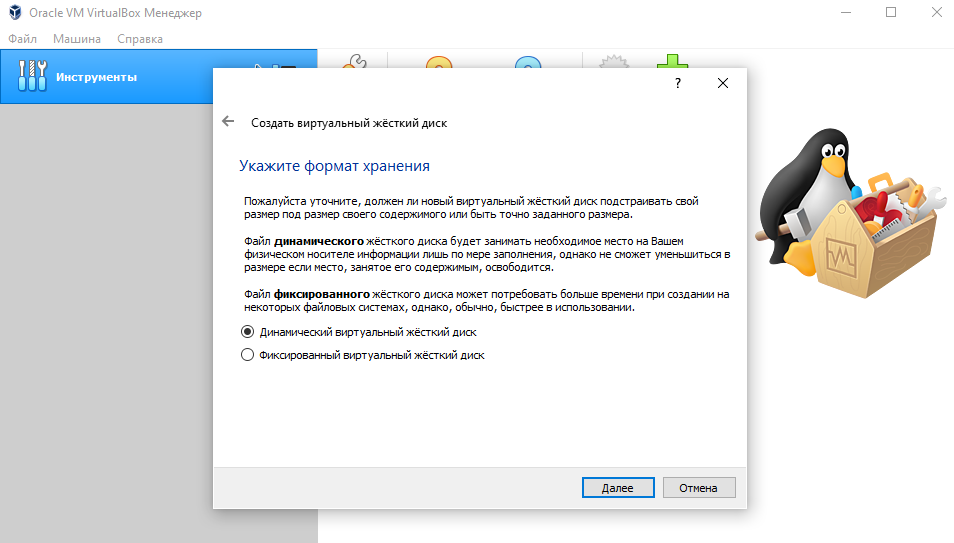


(Рисунок 5)

Задаём конфигурацию жёсткого диска – VDI (BirtualBox Disk Image), динамический виртуальный жёсткий диск (Рисунки 6,7)



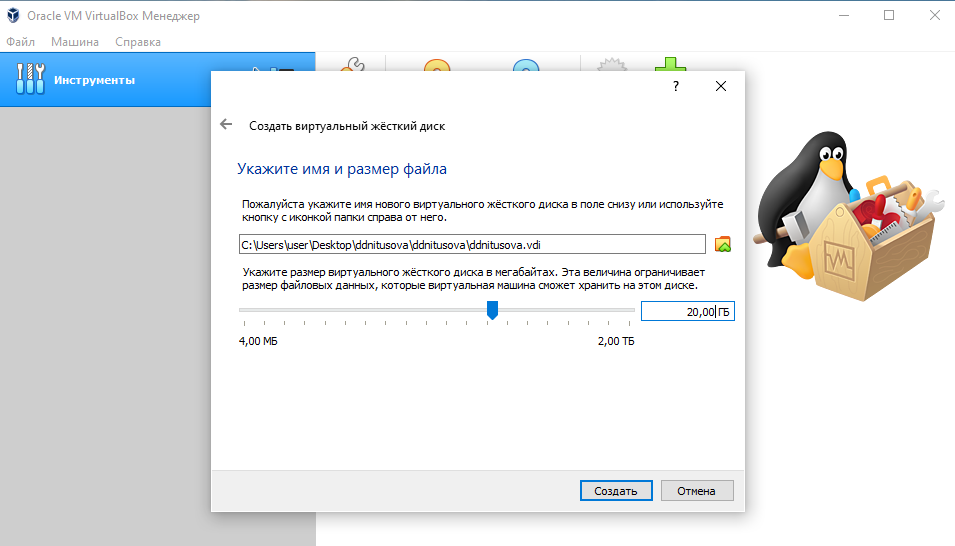
(Рисунок 6)



(Рисунок 7)

Задаём расположение и размер диска. В данном случае:

«C:\Users\user\Desktop\ddnitusova\ddnitusova\ddnitusova.vdi»;20 ГБ (Рисунок 8).

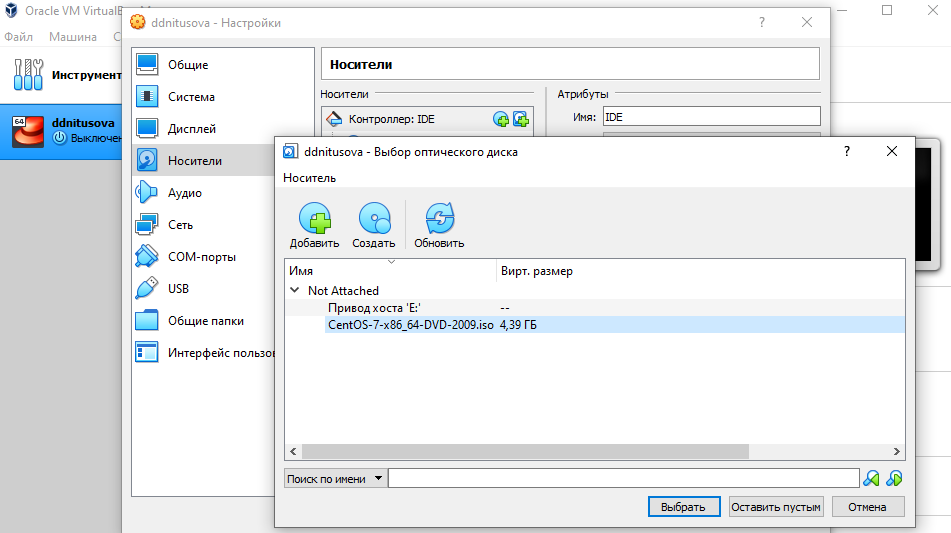


(Рисунок 8)

Теперь в VirtualBox для нашей виртуальной машины выбираем «Свойства» →

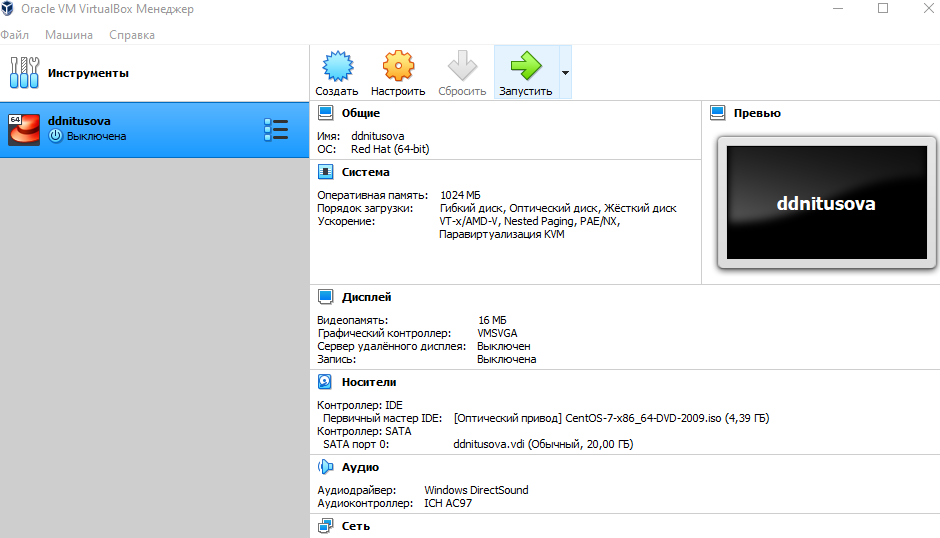
«Носители». Добавляем новый привод оптических дисков и выбираем образ

«CentOS-7-x86\_64-DVD-2009.iso» (Рисунки 9).



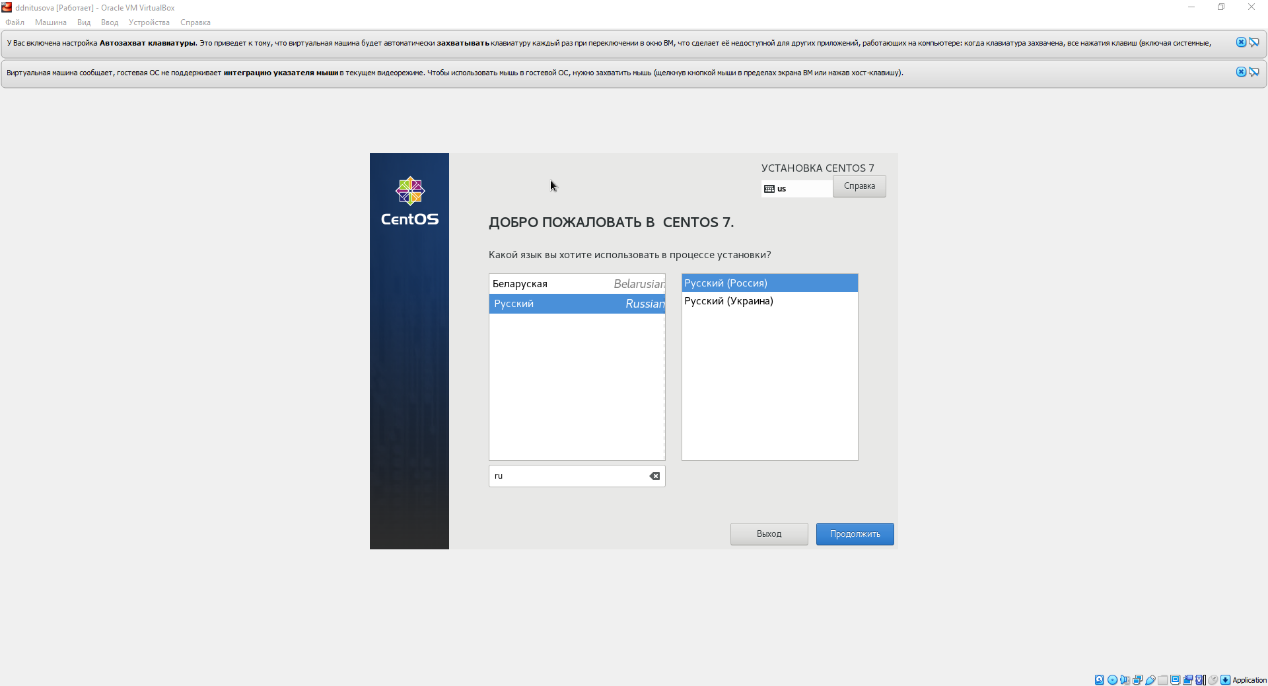
(Рисунок 9)

После этого необходимо запустить виртуальную машину и продолжить настройку (Рисунок 10). В моем случае при обычном запуске возникает «Синий экран смерти» из-зи установленной на моем компьютере программы Кроптопро, поэтому я запустила программу сначала в фоновом режиме и после нажала показать.



(Рисунок 10)

Выбираем язык (Рисунок 11)

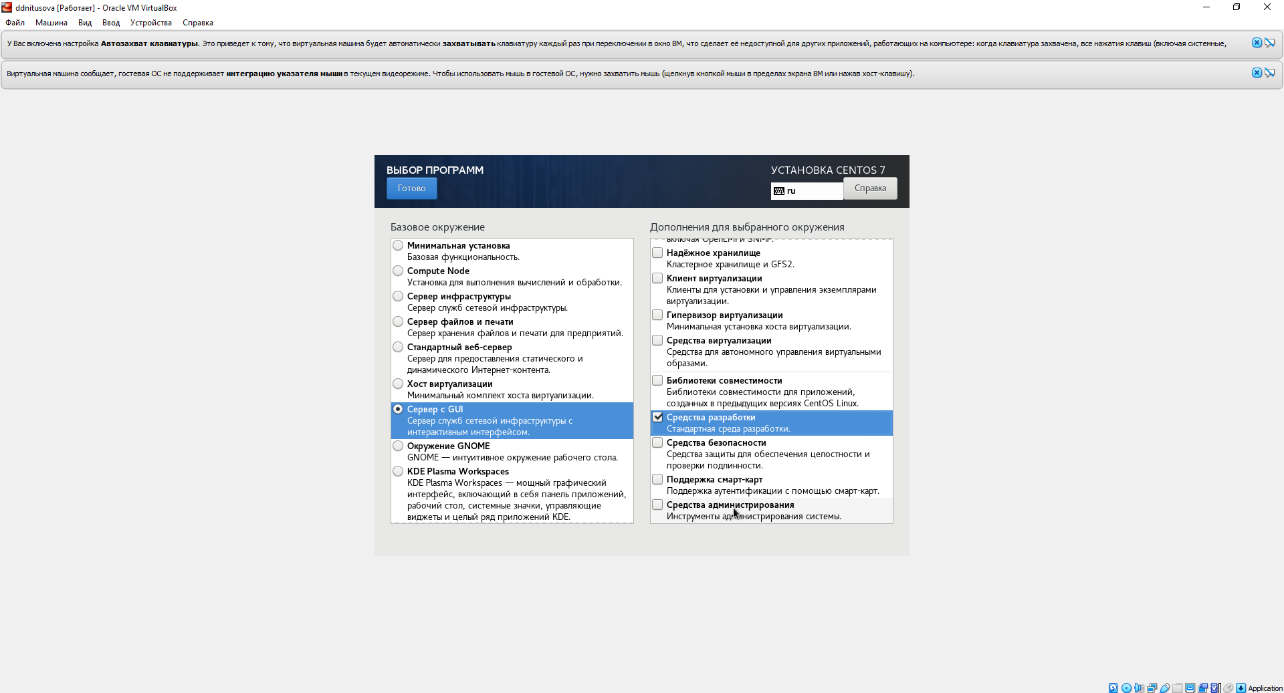


(Рисунок 11)

Настраиваем выбор программ.

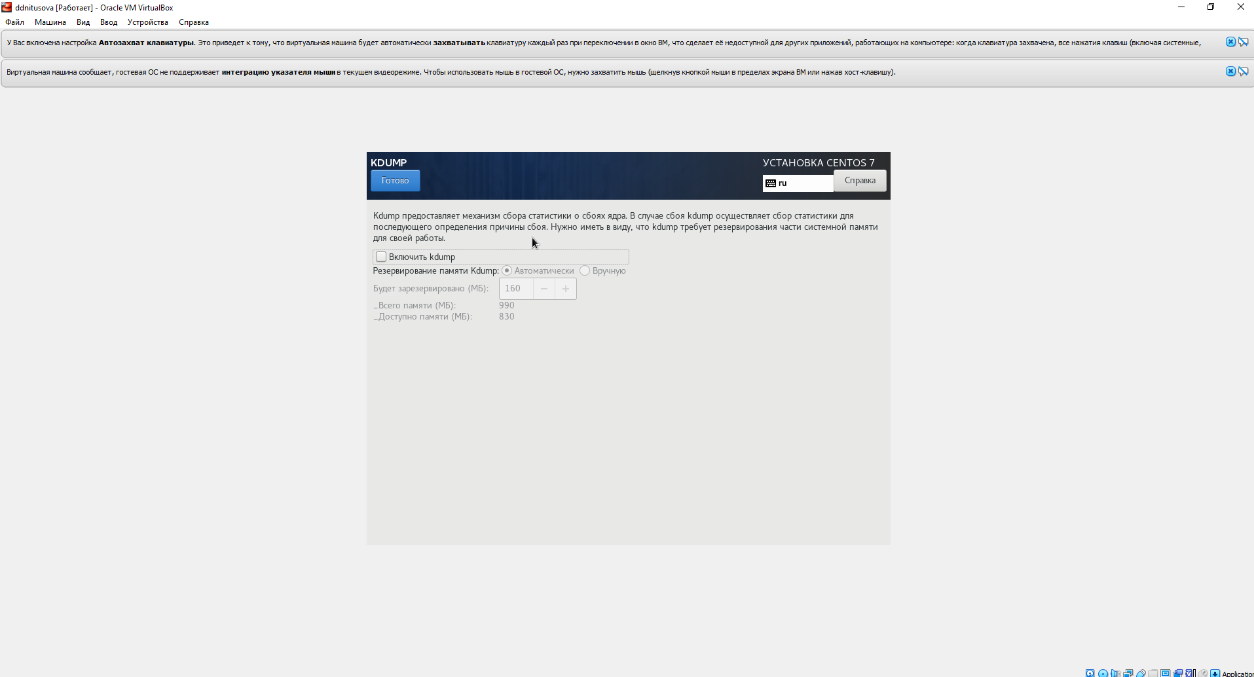
Базовое окружение – Сервер с GUI

Дополнение для окружения – Средства разработки (Рисунок 12)



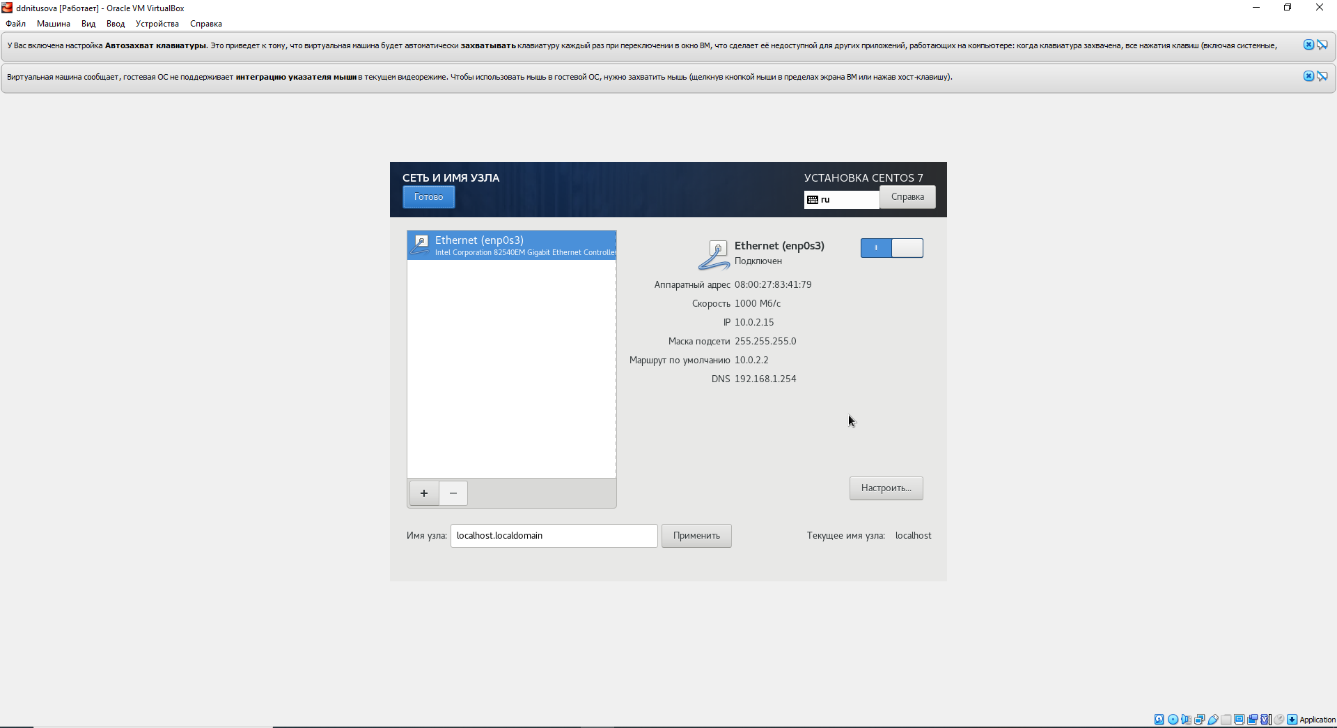
(Рисунок 12)

Отключаем kdump. (Рисунок 13)



(Рисунок 13)

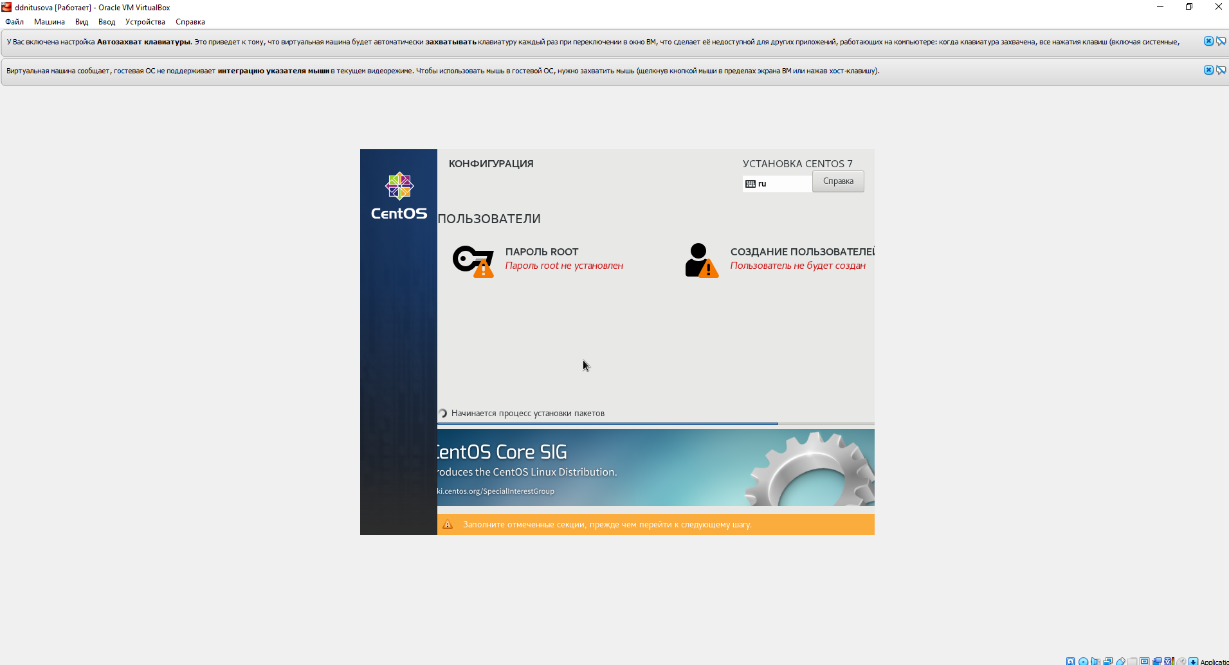
Подключаем сеть и имя узла ddnitusova.localdomain (Рисунок 14)



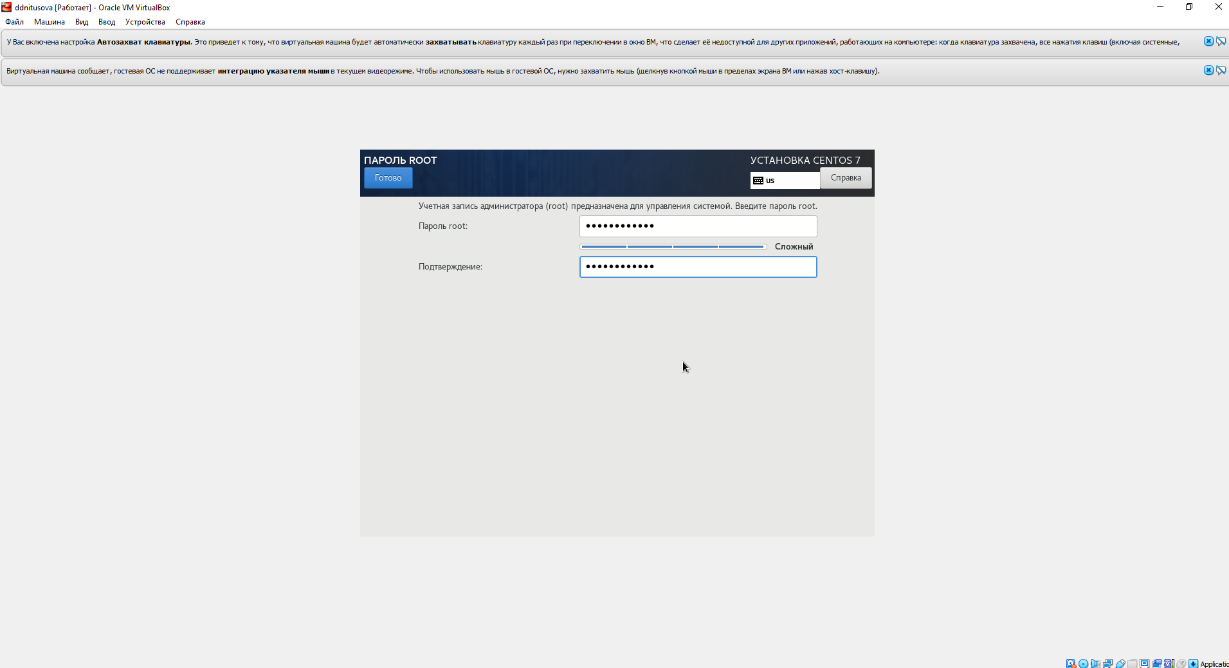
(Рисунок 14)

Нажимаем начать установку.

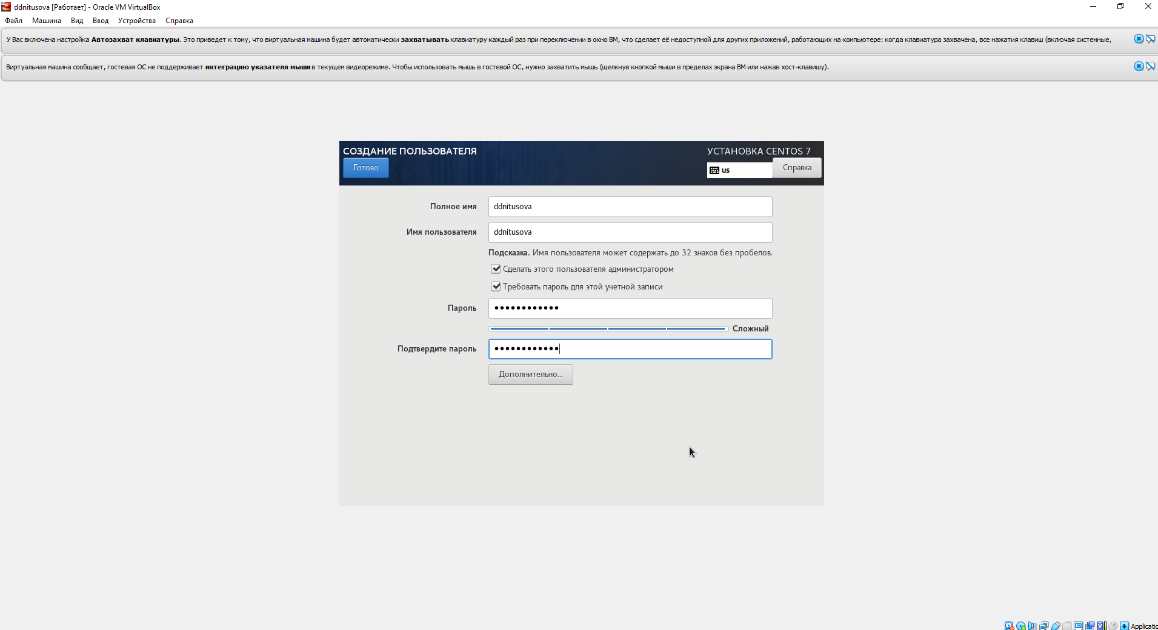
Устанавливаем пароль root и регистрируем пользователя (ddnitusova) и делаем администратором. (Рисунок 15,16,17)



(Рисунок 15)

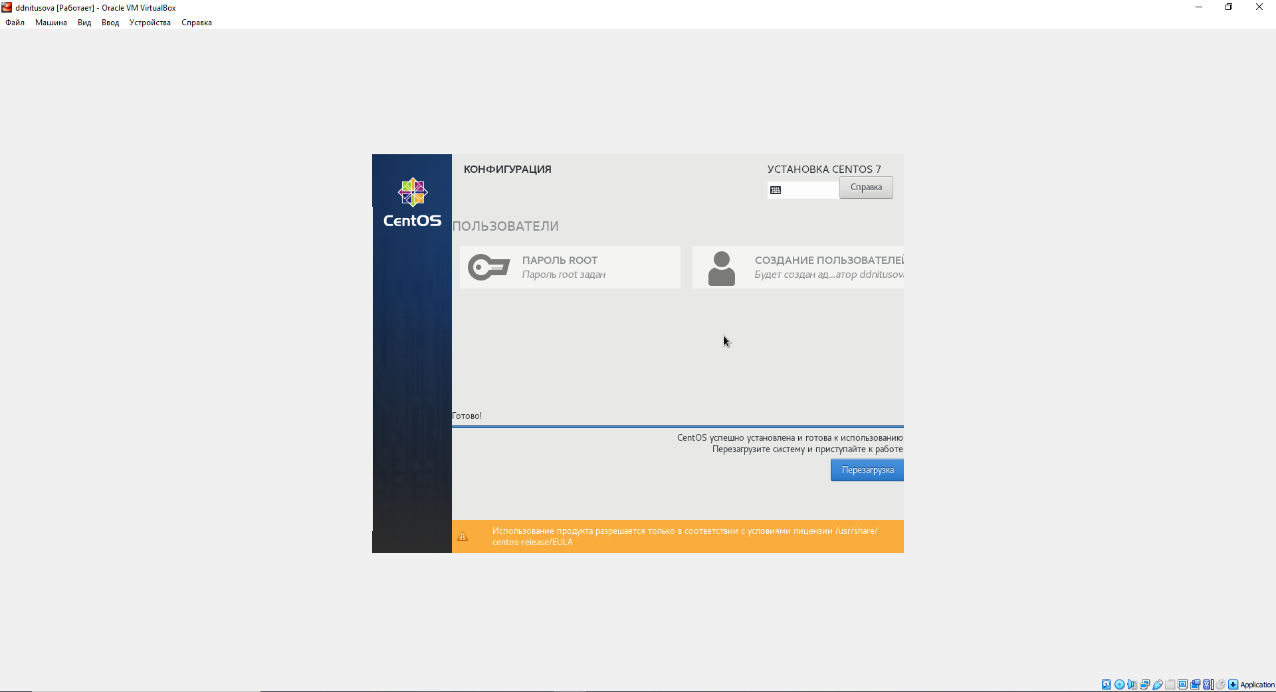


(Рисунок 16)



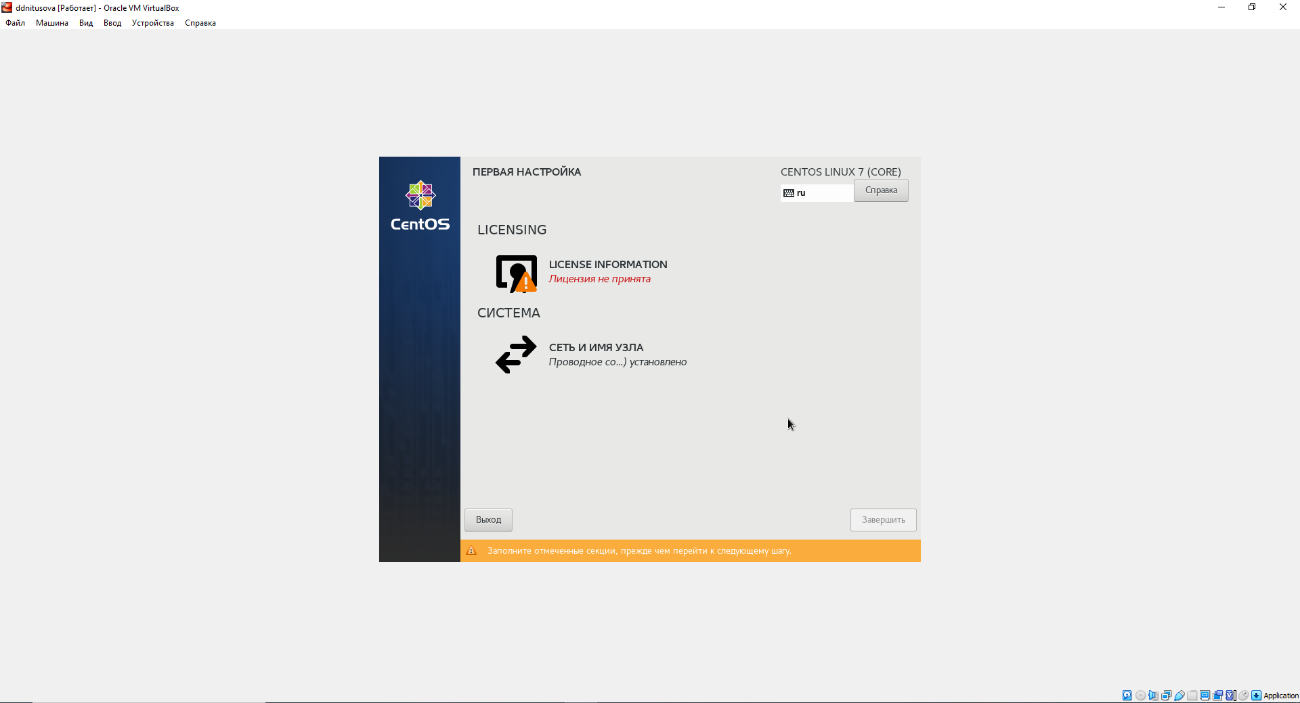
(Рисунок 17)

Ждем, когда установить система и перезагружаем (Рисунок 18)

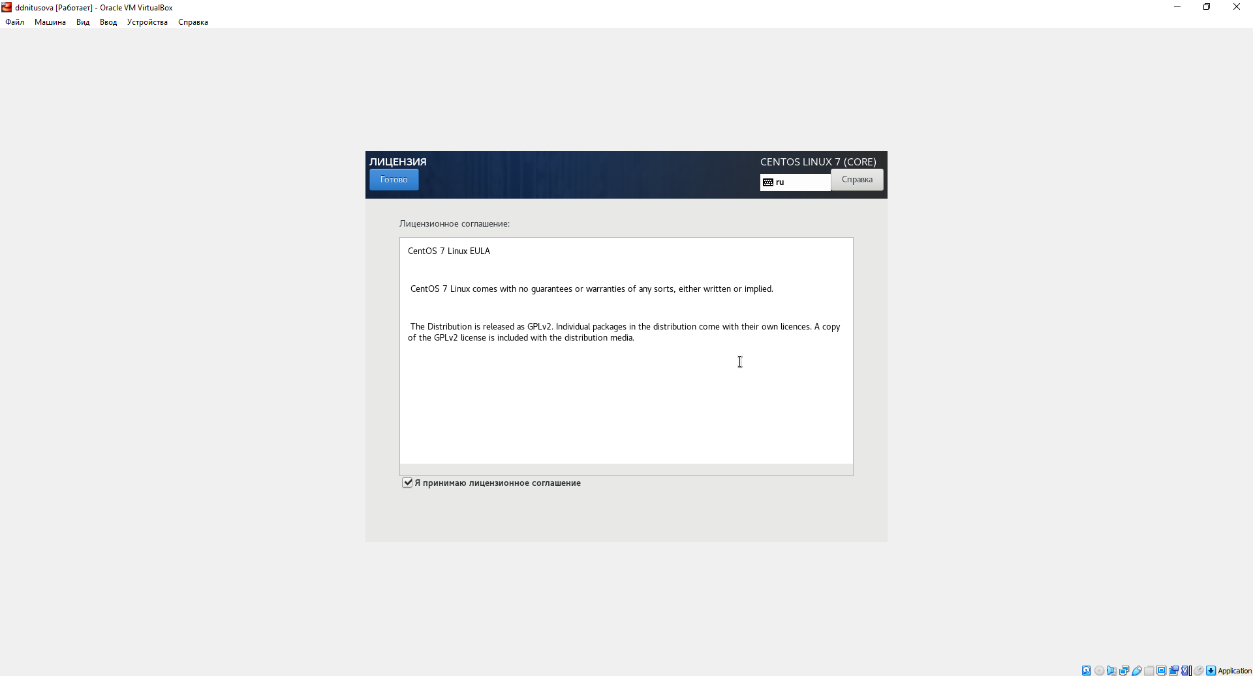


(Рисунок 18)

Принимаем лицензию (Рисунок 19,20)

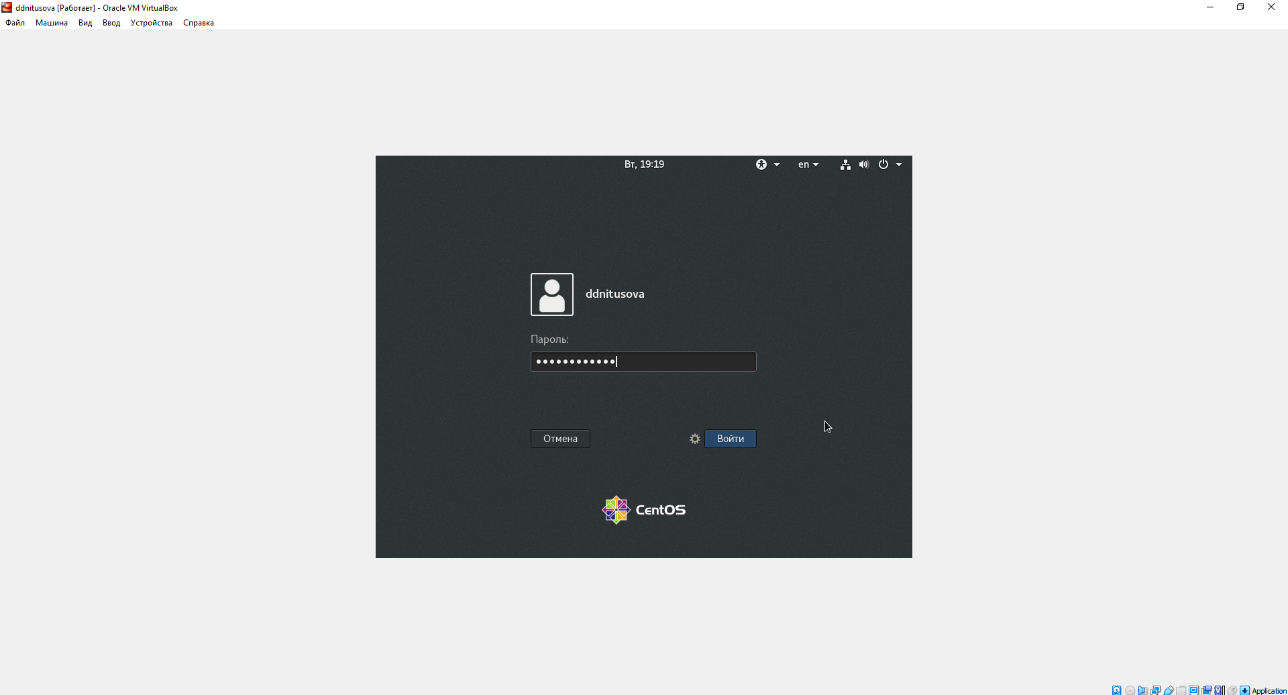


(Рисунок 19)



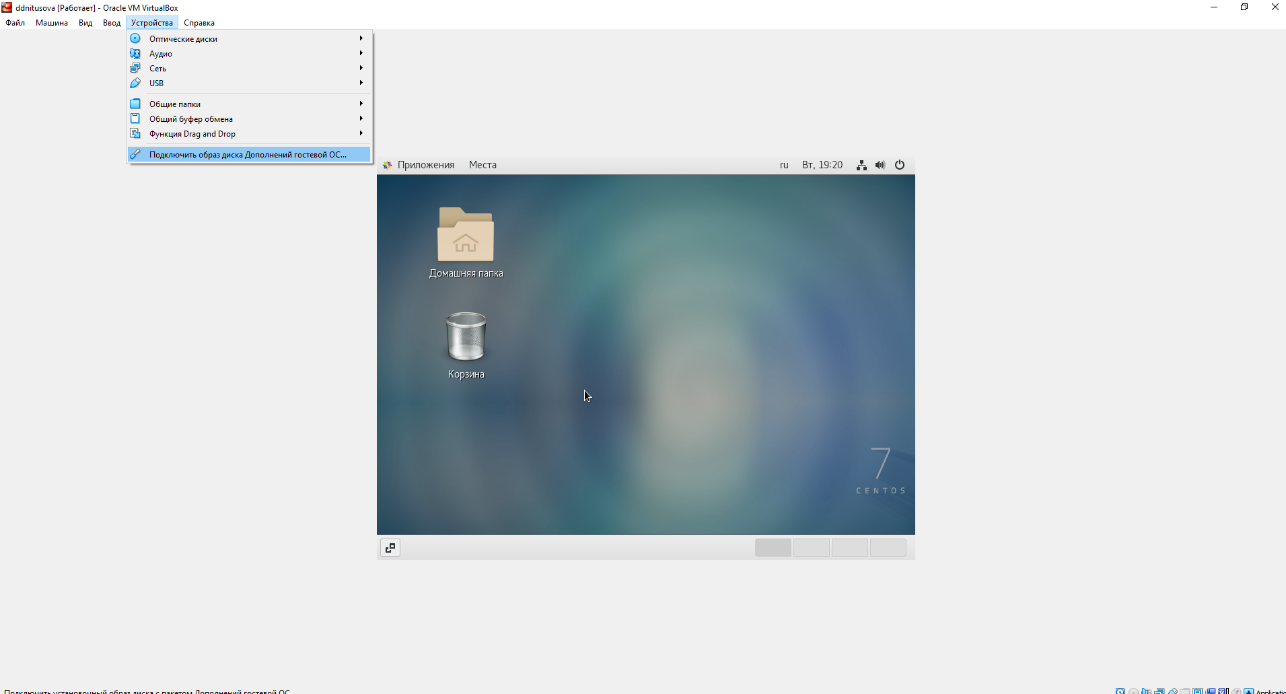
(Рисунок 20)

Перезапускаем систему. Вводим пароль (Рисунок 21)

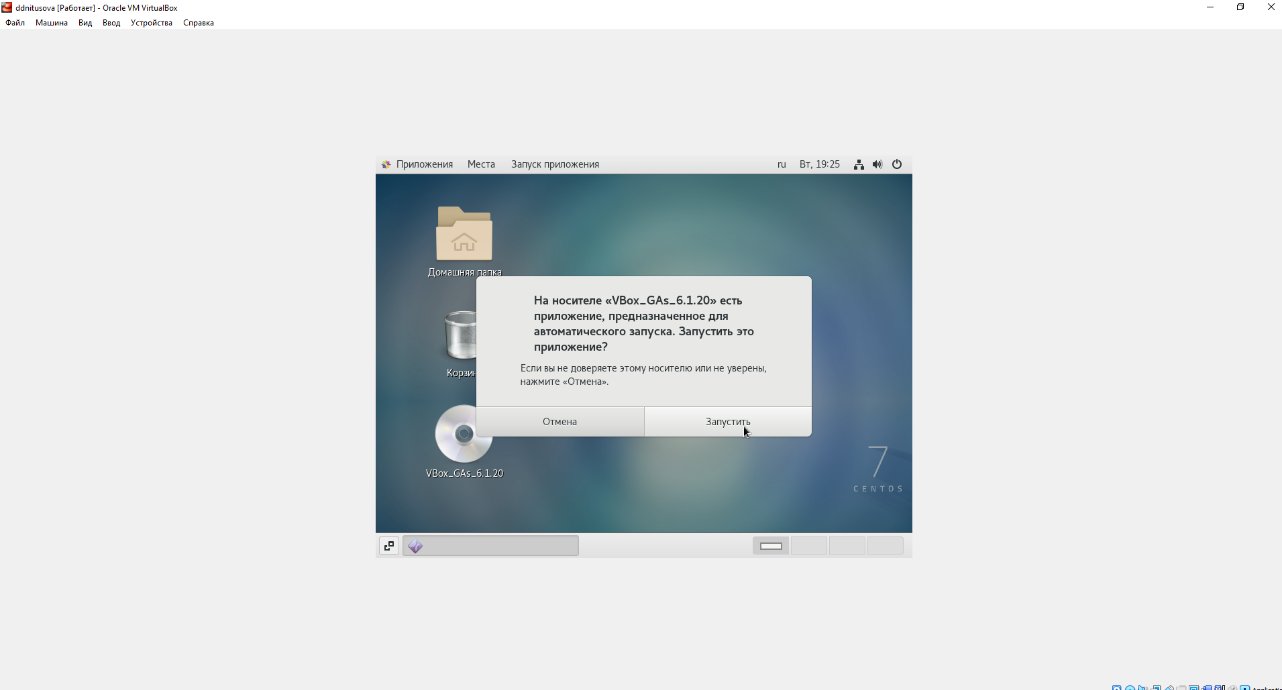


(Рисунок 21)

Подключаем образ доп. системы (драйверы) (Рисунок 22,23)



(Рисунок 22)



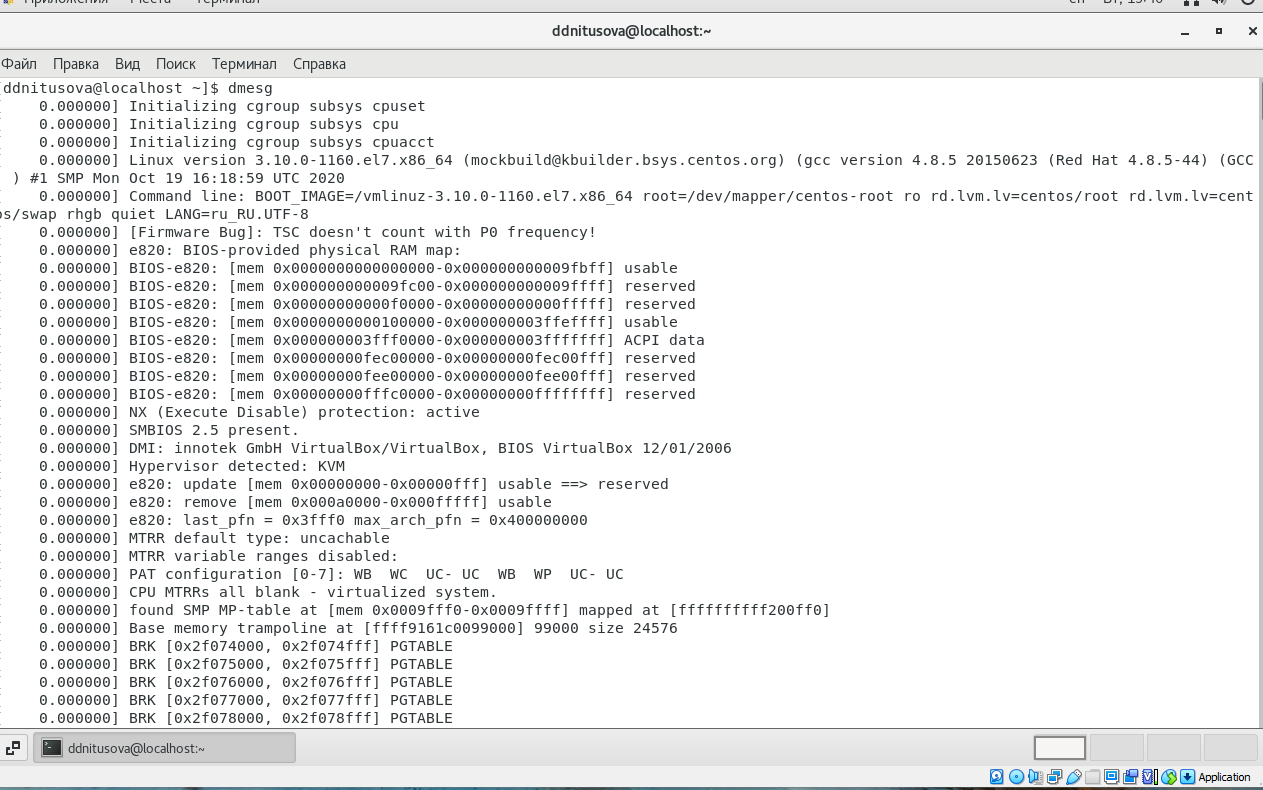
(Рисунок 23)

**Домашнее задание:**

Загружаем графическое окружения и открываем консоль. Анализируем

последовательность загрузки системы, используя команду «dmesg».

(Рисунок 24)



(Рисунок 24)

Далее используем команду «dmesg | grep -i "то, что ищем"», чтобы найти

необходимую информацию.

1) Версия ядра Linux: команда «dmesg | grep -i "Linux version"» (Рисунок 25).

(Рисунок 25)

2) Частота процессора: команда «dmesg | grep –i "MHz"» (Рисунок 25).

3) Модель процессора: команда «dmesg | grep –i "CPU0"» (Рисунок 25).

4) Объем доступной оперативной памяти: команда «dmesg | grep –i

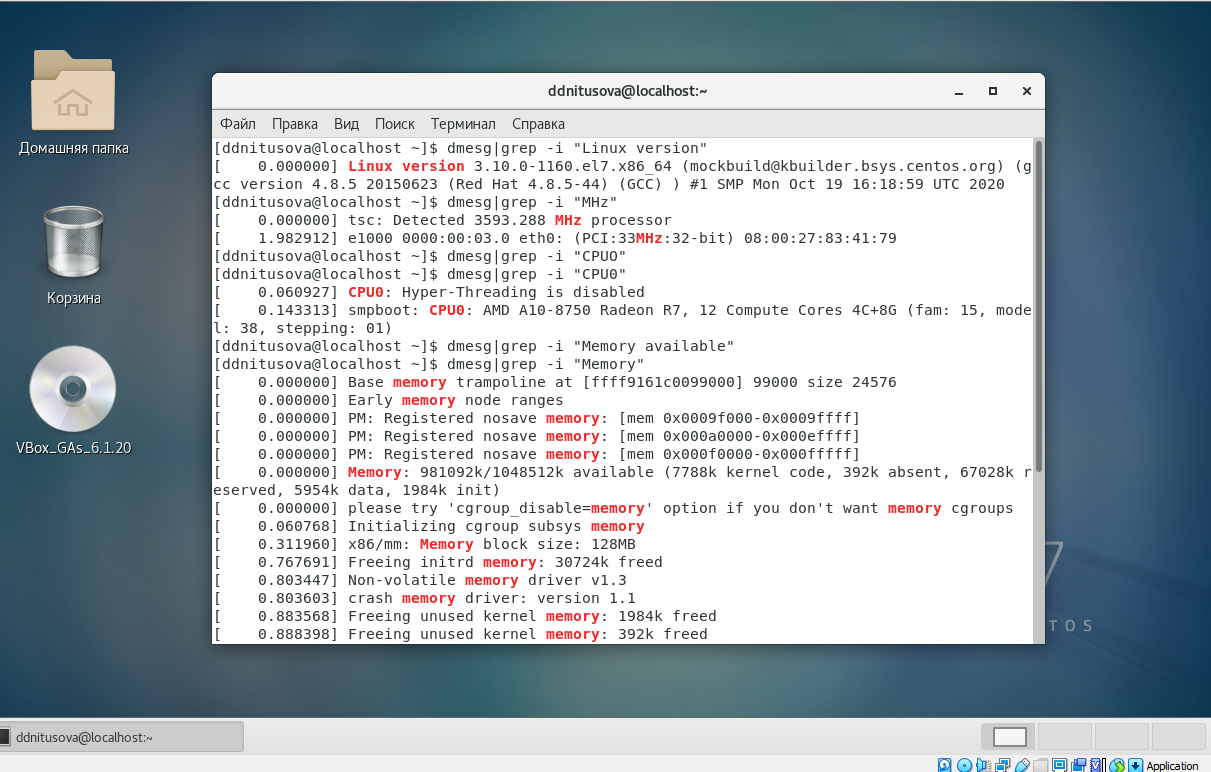
"Memory"» (Рисунок 25,26).

5) Тип обнаруженного гипервизора: команда «dmesg | grep –i

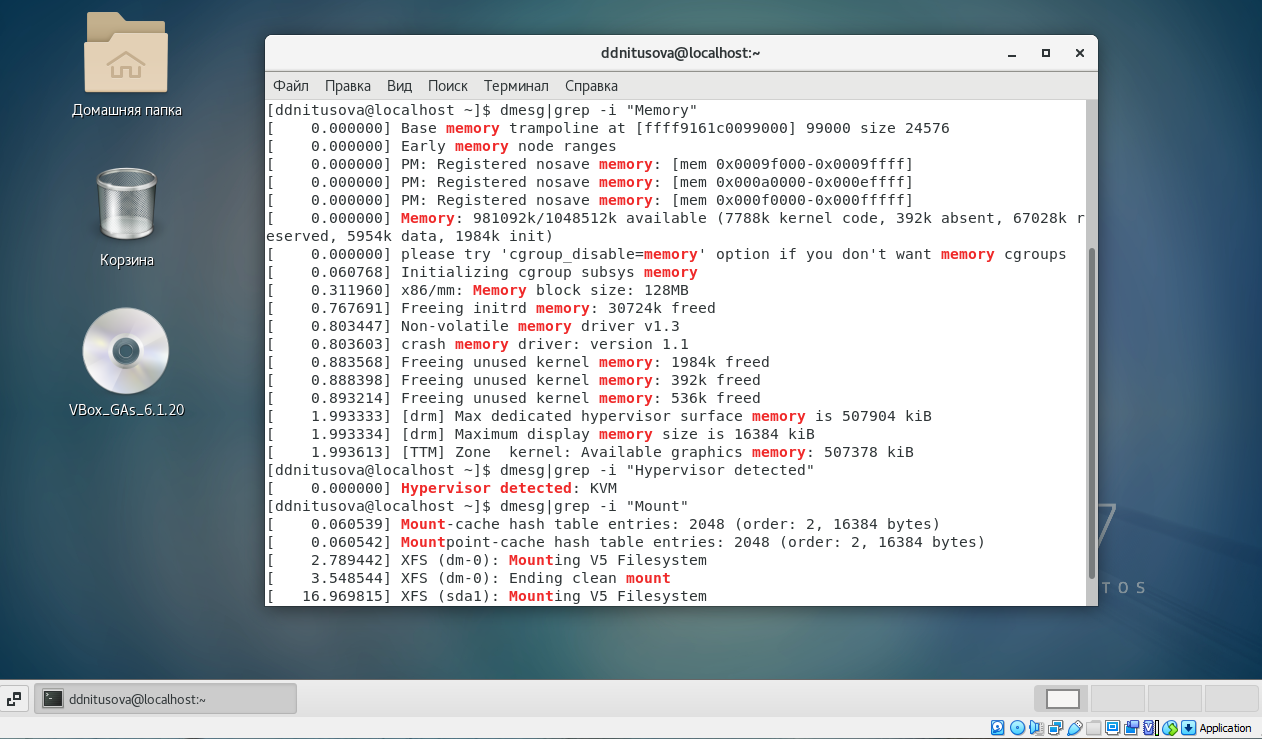
"Hypervisor detected"» (Рисунок 26).

6), 7) Тип файловой системы корневого раздела и последовательность

монтирования файловых систем: команда «dmesg | grep –i "Mount"» (Рисунок 26)



(Рисунок 25)



(Рисунок 26)

**Контрольные вопросы:**

1) Учетная запись пользователя – это необходимая для системы

информация о пользователе, хранящаяся в специальных файлах.

Информация используется Linux для аутентификации пользователя и

назначения ему прав доступа. Аутентификация – системная процедура,

позволяющая Linux определить, какой именно пользователь

осуществляет вход. Вся информация о пользователе обычно хранится в

файлах /etc/passwd и /etc/group.

Учётная запись пользователя содержит:

Имя пользователя (user name)

Идентификационный номер пользователя (UID)

Идентификационный номер группы (GID).

Пароль (password)

Полное имя (full name)

Домашний каталог (home directory)

Начальную оболочку (login shell)

2) Команды терминала:

Для получения справки по команде:

man [команда]. Например, команда «man ls» выведет справку о

команде «ls».

Для перемещения по файловой системе:

cd [путь]. Например, команда «cd newdir» осуществляет переход в

каталог newdir

Для просмотра содержимого каталога:

ls [опции] [путь]. Например, команда «ls –a ~/newdir» отобразит

имена скрытых файлов в каталоге newdir

Для определения объёма каталога:

du [опция] [путь]. Например, команда «du –k ~/newdir» выведет

размер каталога newdir в килобайтах

 Для создания / удаления каталогов / файлов:

mkdir [опции] [путь] / rmdir [опции] [путь] / rm [опции] [путь].

Например, команда «mkdir –p ~/newdir1/newdir2» создаст

иерархическую цепочку подкаталогов, создав каталоги newdir1 и

newdir2; команда «rmdir -v ~/newdir» удалит каталог newdir;

команда «rm –r ~/newdir» так же удалит каталог newdir

Для задания определённых прав на файл / каталог:

сhmod [опции] [путь]. Например, команда «сhmod g+r ~/text.txt»

даст группе право на чтение файла text.txt

Для просмотра истории команд:

history [опции]. Например, команда «history 5» покажет список

последних 5 команд

3) Файловая система имеет два значения: с одной стороны – это

архитектура хранения битов на жестком диске, с другой – это

организация каталогов в соответствии с идеологией Unix.

Файловая система (англ. «file system») – это архитектура хранения

данных в системе, хранение данных в оперативной памяти и доступа к

конфигурации ядра. Файловая система устанавливает физическую и

логическую структуру файлов, правила их создания и управления ими.

В физическом смысле файловая система Linux представляет собой

пространство раздела диска, разбитое на блоки фиксированного

размера. Их размер кратен размеру сектора: 1024, 2048, 4096 или 8120

байт.

Существует несколько типов файловых систем:

XFS – начало разработки 1993 год, фирма Silicon Graphics, в мае

2000 года предстала в GNU GPL, для пользователей большинства

Linux систем стала доступна в 2001-2002 гг. Отличительная черта

системы – прекрасная поддержка больших файлов и файловых

томов, 8 эксбибайт (8\*260 байт) для 64-х битных систем.

ReiserFS (Reiser3) – одна из первых журналируемых файловых

систем под Linux, разработана Namesys, доступна с 2001 г.

Максимальный объём тома для этой системы равен 16 тебибайт

(16\*240 байт).

JFS (Journaled File System) – файловая система, детище IBM,

явившееся миру в далёком 1990 году для ОС AIX (Advanced

Interactive eXecutive). В виде первого стабильного релиза, для

пользователей Linux, система стала доступна в 2001 году. Из

плюсов системы – хорошая масштабируемость. Из минусов – не

особо активная поддержка на протяжении всего жизненного цикла.

Максимальный рамер тома 32 пэбибайта (32\*250 байт).

ext (extended filesystem) – появилась в апреле 1992 года, это была

первая файловая система, изготовленная специально под нужды

Linux ОС. Разработана Remy Card с целью преодолеть ограничения

файловой системы Minix.

ext2 (second extended file system) – была разработана Remy Card в

1993 году. Не журналируемая файловая система, это был основной

её недостаток, который исправит ext3.

ext3 (third extended filesystem) – по сути расширение исконной для

Linux ext2, способное к журналированию. Разработана Стивеном

Твиди (Stephen Tweedie) в 1999 году, включена в основное ядро

Linux в ноябре 2001 года. На фоне других своих сослуживцев

обладает более скромным размером пространства, до 4 тебибайт

(4\*240 байт) для 32-х разрядных систем. На данный момент является

наиболее стабильной и поддерживаемой файловой системой в среде

Linux.

Reiser4 – первая попытка создать файловую систему нового

поколения для Linux. Впервые представленная в 2004 году, система

включает в себя такие передовые технологии как транзакции,

задержка выделения пространства, а также встроенная

возможность кодирования и сжатия данных. Ханс Рейзер (Hans

Reiser) – главный разработчик системы.

ext4 – попытка создать 64-х битную ext3 способную поддерживать

больший размер файловой системы (1 эксбибайт). Позже

добавились возможности – непрерывные области дискового

пространства, задержка выделения пространства, онлайн

дефрагментация и прочие. Обеспечивается прямая совместимость с

системой ext3 и ограниченная обратная совместимость при

недоступной способности к непрерывным областям дискового

пространства.

Btrfs (B-tree FS или Butter FS) – проект изначально начатый

компанией Oracle, впоследствии поддержанный большинством

Linux систем. Ключевыми особенностями данной файловой

системы являются технологии: copy-on-write, позволяющая сделать

снимки областей диска (снапшоты), которые могут пригодится для

последующего восстановления; контроль за целостностью данных и

метаданных (с повышенной гарантией целостности); сжатие

данных; оптимизированный режим для накопителей SSD (задаётся

при монтировании) и прочие. Немаловажным фактором является

возможность перехода с ext3 на Btrfs. С августа 2008 года данная

система выпускается под GNU GPL.

Tux2 – известная, но так и не анонсированная публично файловая

система. Создатель Дэниэл Филипс (Daniel Phillips). Cистема

базируется на алгоритме «Фазового Дерева», который, как и

журналирование защищает файловую систему от сбоев.

Организована как надстройка на ext2.

Tux3 – cистема создана на основе FUSE (Filesystem in Userspace),

специального модуля для создания файловых систем на Unix

платформах. Данный проект ставит перед собой цель избавиться от

привычного журналирования, взамен предлагая версионное

восстановление (состояние в определённый промежуток времени).

Преимуществом используемой в данном случае версионной

системы, является способ описания изменений, где для каждого

файла создаётся изменённая копия, а не переписывается текущая

версия.

Xiafs – задумка и разработка данной файловой системы

принадлежат Frank Xia, основана на файловой системе MINIX. В

настоящее время считается устаревшей и практически не

используется. Наряду с ext2 разрабатывалась, как замена системе

ext. В декабре 1993 года система была добавлена в стандартное

ядро Linux. И хотя система обладала большей стабильностью и

занимала меньше дискового пространства под контрольные

структуры – она оказалась слабее ext2, ведущую роль сыграли

ограничения максимальных размеров файла и раздела, а также

способность к дальнейшему расширению.

ZFS (Zettabyte File System) – изначально созданная в Sun

Microsystems файловая система, для небезызвестной операционной

системы Solaris в 2005 году. Отличительные особенности –

отсутствие фрагментации данных как таковой, возможности по

управлению снапшотами (snapshots), пулами хранения (storage

pools), варьируемый размер блоков, 64-х разрядный механизм

контрольных сумм, а также способность адресовать 128 бит

информации. В Linux системах может использоваться посредствам

FUSE.

4) Команда «findmnt» или «findmnt --all» будет отображать все

подмонтированные файловые системы или искать файловую систему.

5) Основные сигналы (каждый сигнал имеет свой номер), которые

используются для завершения процесса:

SIGINT – самый безобидный сигнал завершения, означает Interrupt.

Он отправляется процессу, запущенному из терминала с помощью

сочетания клавиш Ctrl+C. Процесс правильно завершает все свои

действия и возвращает управление;

SIGQUIT – это еще один сигнал, который отправляется с помощью

сочетания клавиш, программе, запущенной в терминале. Он

сообщает ей что нужно завершиться и программа может выполнить

корректное завершение или проигнорировать сигнал. В отличие от

предыдущего, она генерирует дамп памяти. Сочетание клавиш

Ctrl+/;

SIGHUP – сообщает процессу, что соединение с управляющим

терминалом разорвано, отправляется, в основном, системой при

разрыве соединения с интернетом;

SIGTERM – немедленно завершает процесс, но обрабатывается

программой, поэтому позволяет ей завершить дочерние процессы и

освободить все ресурсы;

SIGKILL – тоже немедленно завершает процесс, но, в отличие от

предыдущего варианта, он не передается самому процессу, а

обрабатывается ядром. Поэтому ресурсы и дочерние процессы

остаются запущенными.

Также для передачи сигналов процессам в Linux используется утилита

kill, её синтаксис: kill [-сигнал] [pid\_процесса] (PID – уникальный

идентификатор процесса). Сигнал представляет собой один из выше

перечисленных сигналов для завершения процесса.

Перед тем, как выполнить остановку процесса, нужно определить его

PID. Для этого используют команды ps и grep. Команда

ps предназначена для вывода списка активных процессов в системе

и информации о них. Команда grep запускается одновременно с ps

(в канале) и будет выполнять поиск по результатам команды ps.

Утилита pkill – это оболочка для kill, она ведет себя точно так же, и

имеет тот же синтаксис, только в качестве идентификатора процесса ей

нужно передать его имя.

killall работает аналогично двум предыдущим утилитам. Она тоже

принимает имя процесса в качестве параметра и ищет его PID в

директории /proc. Но эта утилита обнаружит все процессы с таким

именем и завершит их.

**Вывод:** Я приобрела практические навыки установки операционной системы на виртуальную машину и научилась настраивать для дальнейшей работы сервисов.