

**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: Дарижапов Тимур Андреевич

Группа: НПМбд-01-20

**МОСКВА**

2020 г.

**Цель работы:** приобретение практических навыков установки операционной системы на виртуальную машину, настройки минимально необходимых для дальнейшей работы сервисов.

**Ход работы: 1.** Проверим в свойствах VirtualBox месторасположение каталога для виртуальных машин. В поле Папка для машин (рис.1) должно стоять /var/tmp/имя\_пользователя. В моём случае – tadarizhapov. Если указан другой каталог, то требуется изменить его.

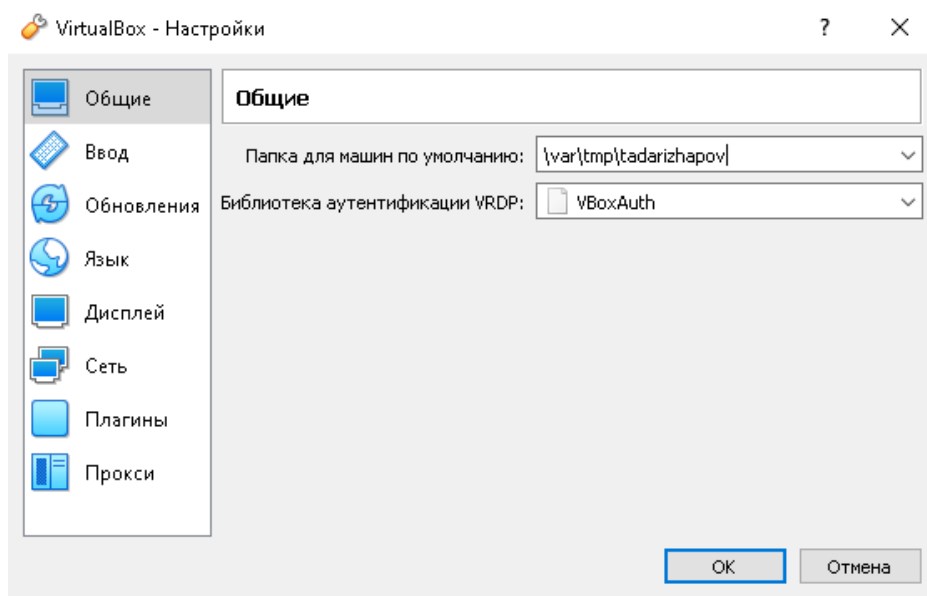


рис.1

2. Создаём новую виртуальную машину. Имя – tadarizhapov. Система Linux. Версия – Red Hat.

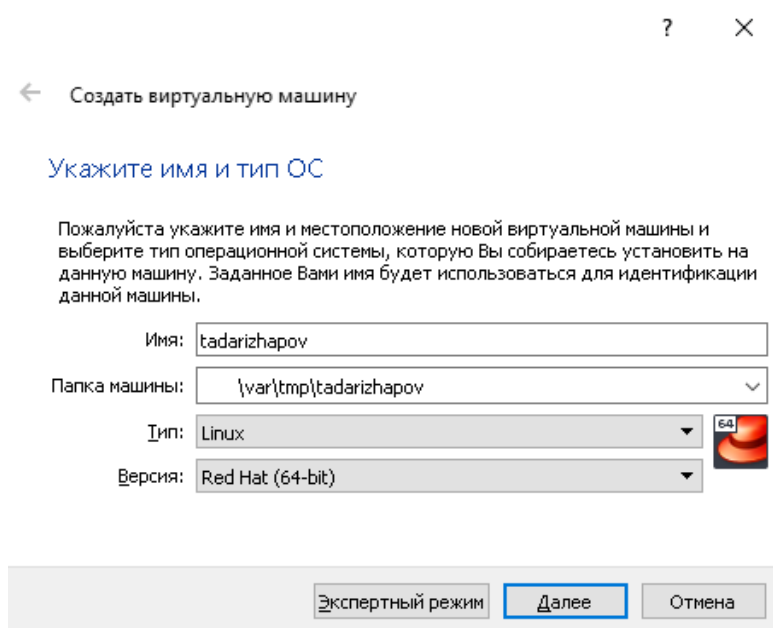


рис.2

### 3. Объём памяти. В моём случае – 1 ГБ.

Создать виртуальную машину

#### Укажите объём памяти

Укажите объём оперативной памяти (RAM) выделенный данной виртуальной машине.

Рекомендуемый объём равен **1024** МБ.

4 МБ 4096 МБ

1024 МБ

Далее Отмена

рис.3

### 4. Задаём конфигурацию жёсткого диска. Новый виртуальный жёсткий диск

Создать виртуальную машину

#### Жесткий диск

При желании к новой виртуальной машине можно подключить виртуальный жёсткий диск. Вы можете создать новый или выбрать из уже имеющихся.

Если Вам необходима более сложная конфигурация Вы можете пропустить этот шаг и внести изменения в настройки машины после её создания.

Рекомендуемый объём нового виртуального жёсткого диска равен **8,00** ГБ.

☐ Не подключать виртуальный жёсткий диск

☒ Создать новый виртуальный жёсткий диск

☐ Использовать существующий виртуальный жёсткий диск

LinuxUbuntu.vdi (Обычный, 11,00 ГБ)

Создать Отмена

рис.4

Тип – VDI(VirtualBox Disk Image).

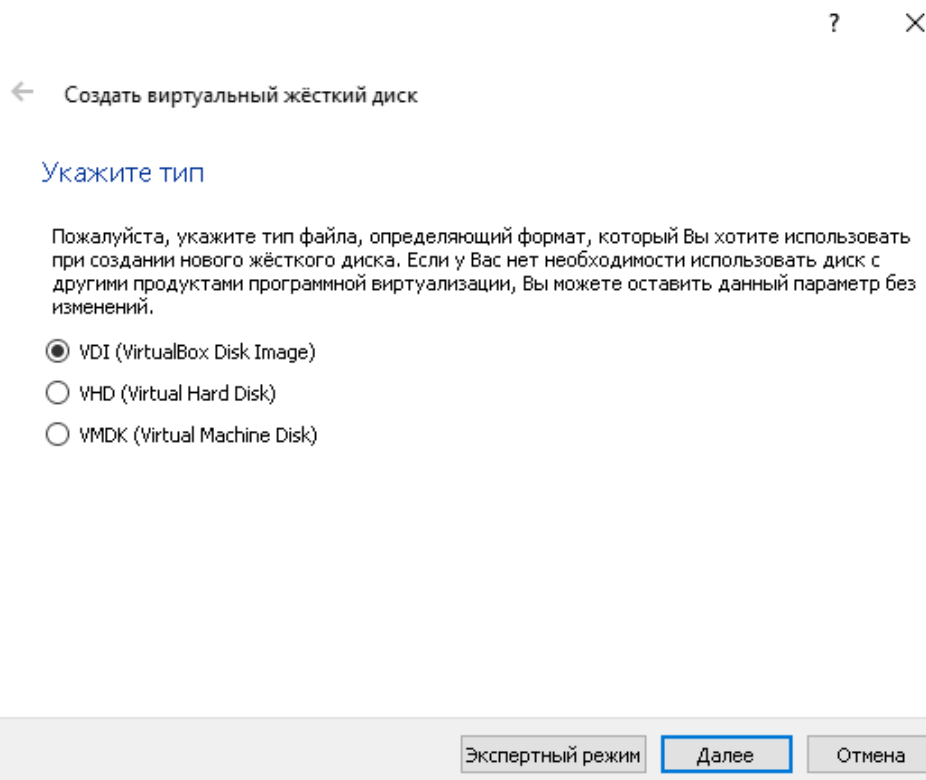


рис.5

## Формат хранения – Динамический жёсткий диск.

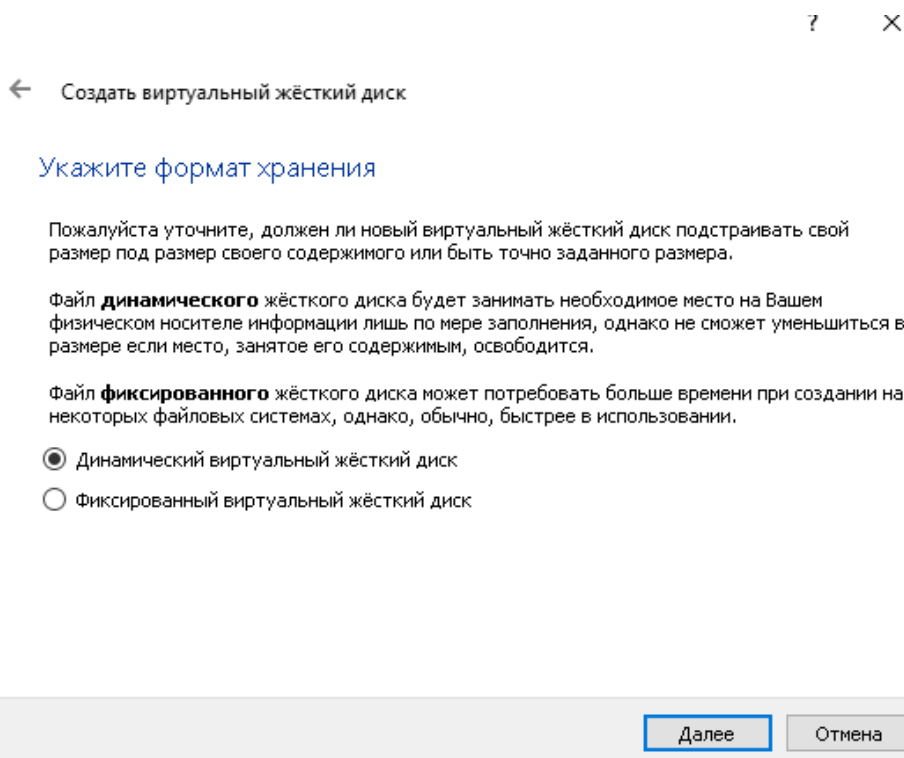


рис.6

7. Укажем имя и размер файла. Желательно использовать 40 ГБ и больше.

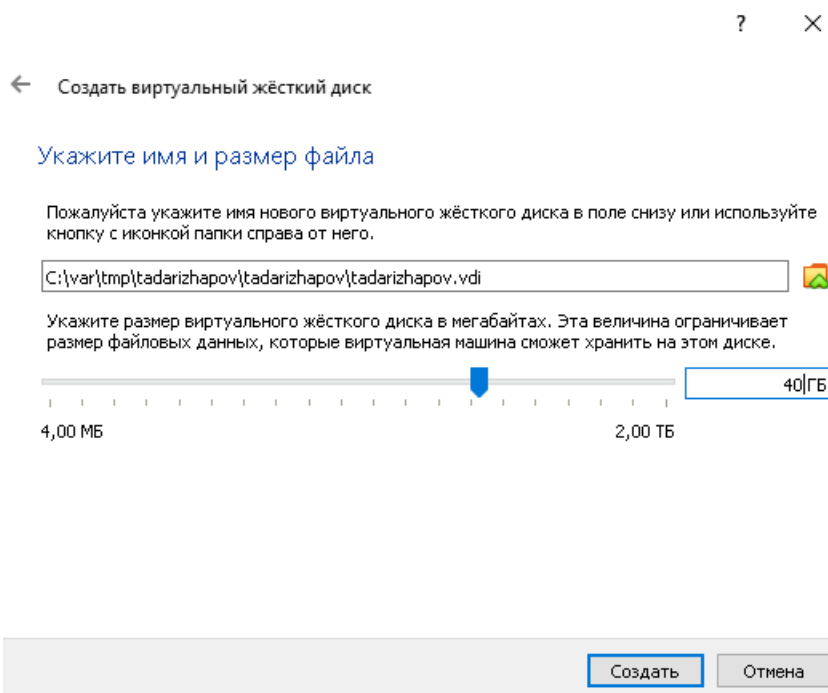


рис.7

8.Заходим в Свойства – Носители. Добавляем новый привод оптических дисков и выбираем образ CentOS-7-x86. После запускаем виртуальную машину.

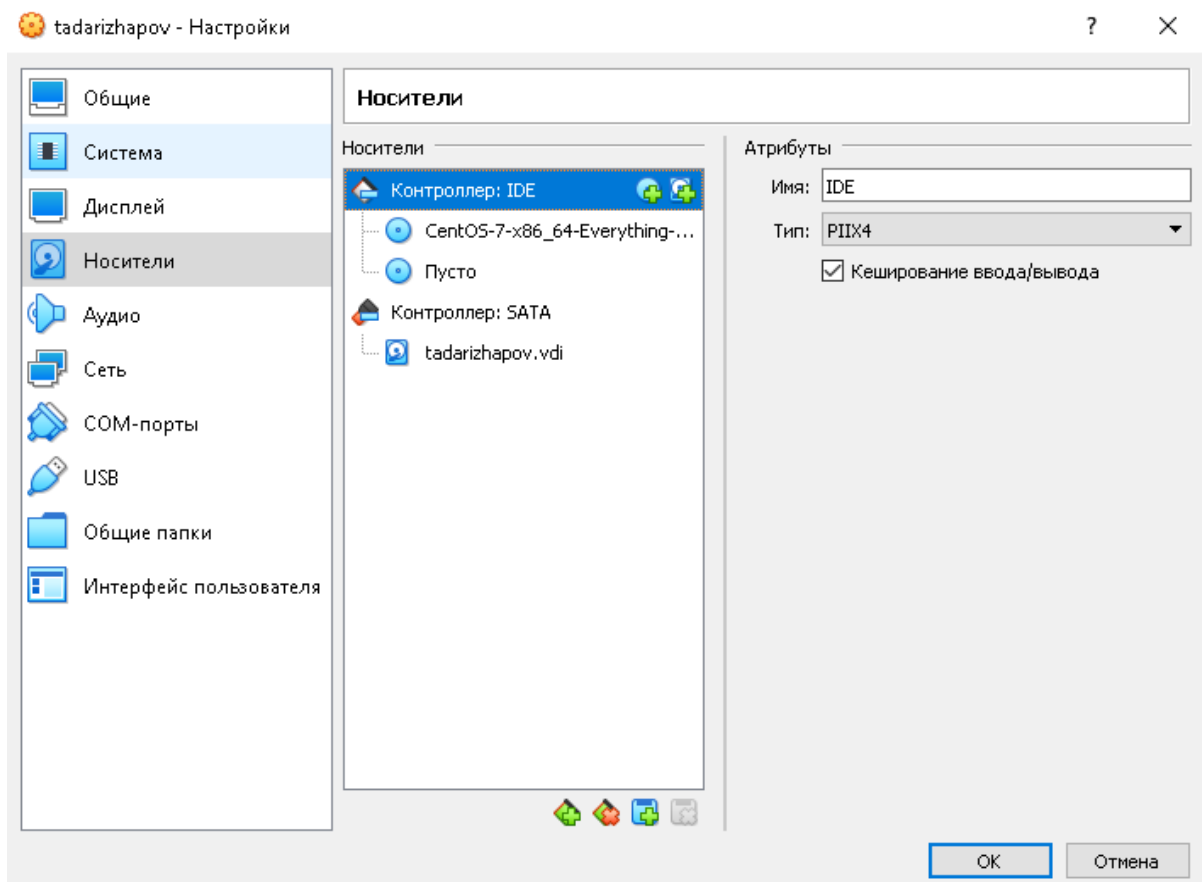


рис.8

Нас встречает загрузка.

```
[ 9.498118] dracut-pre-udev[329]: modprobe: ERROR: could not insert 'floppy':
No such device
[ OK ] Started Show Plymouth Boot Screen.
[ OK ] Reached target Paths.
[ OK ] Started Forward Password Requests to Plymouth Directory Watch.
[ OK ] Reached target Basic System.
[ OK ] Started Device-Mapper Multipath Device Controller.
Starting Open-iSCSI...
[ OK ] Started Open-iSCSI.
Starting dracut initqueue hook...
[ 11.875732] [drm:vmw_host_log [vmwgfx]] *ERROR* Failed to send host log messa
ge.
[ 11.877520] [drm:vmw_host_log [vmwgfx]] *ERROR* Failed to send host log messa
ge.
[ 13.654381] dracut-initqueue[714]: mount: /dev/sr0 is write-protected, mounting read-only
[ OK ] Started Show Plymouth Boot Screen.
[ OK ] Reached target Paths.
[ OK ] Started Forward Password Requests to Plymouth Directory Watch.
[ OK ] Reached target Basic System.
[ OK ] Started Device-Mapper Multipath Device Controller.
Starting Open-iSCSI...
[ OK ] Started Open-iSCSI.
Starting dracut initqueue hook...
[ 13.654381] dracut-initqueue[714]: mount: /dev/sr0 is write-protected, mounting read-only
[ OK ] Created slice system-checkisond5.slice.
Starting Media check on /dev/sr0...
/dev/sr0: a3791275dfe9ccac895d502325b123af
Fragment sums: ef932a93979c3418b86cc1c6acd6339c1dbd76f321ff9fcb1df9d48951b7
Fragment count: 20
Press [Esc] to abort check.
Checking: 001.5%_
```

рис.9

9.В окне настройки установки находим выбор программ. Выбираем Сервер с GUI и Средства разработки.

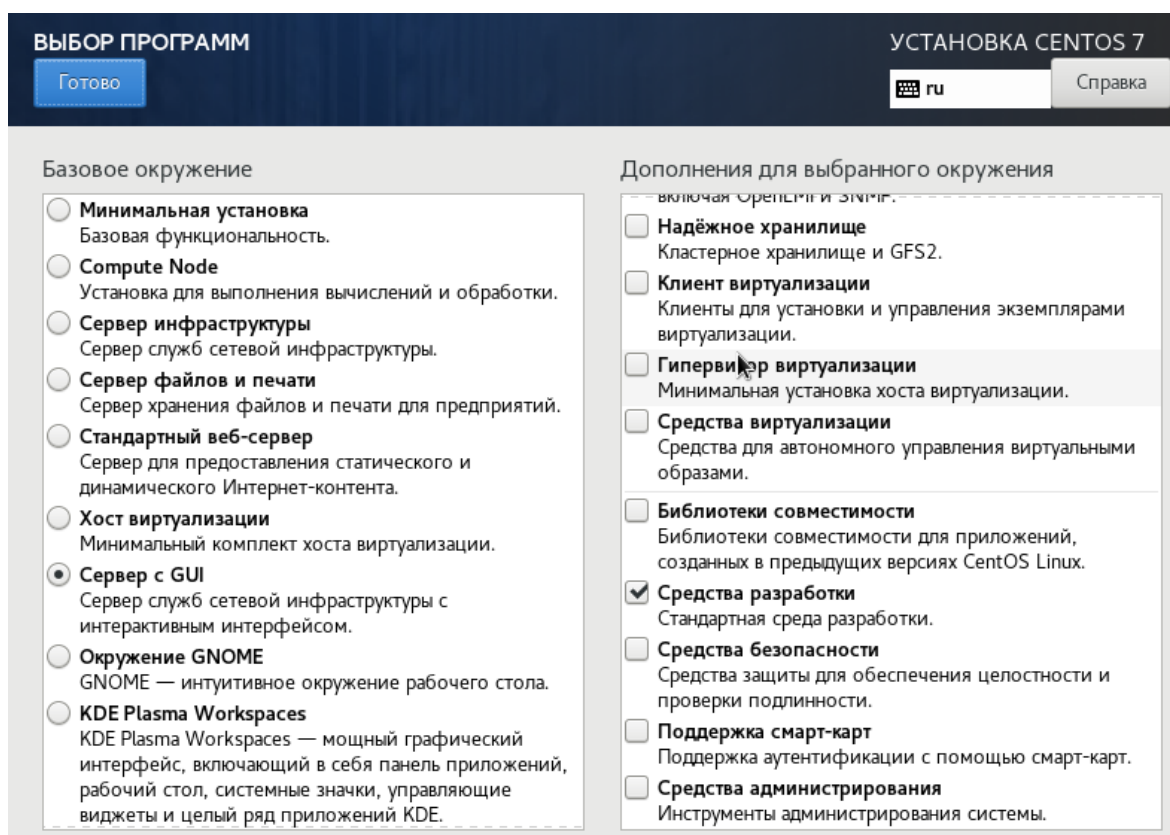


рис.10

## 10. Включаем KDUMP.

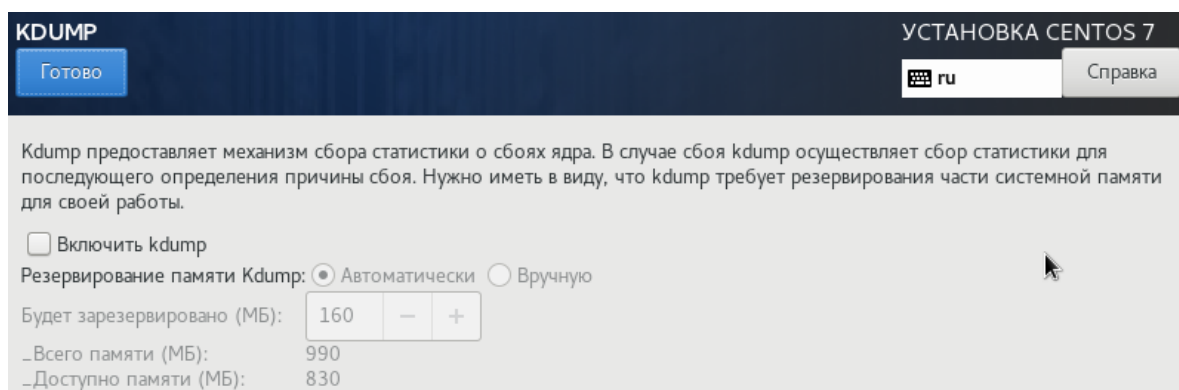


рис.11

11. Места установки не трогаем. Заходим в Сеть и имя узла. Включаем интернет. Имя узла – `tadarizhapov.localdomain`

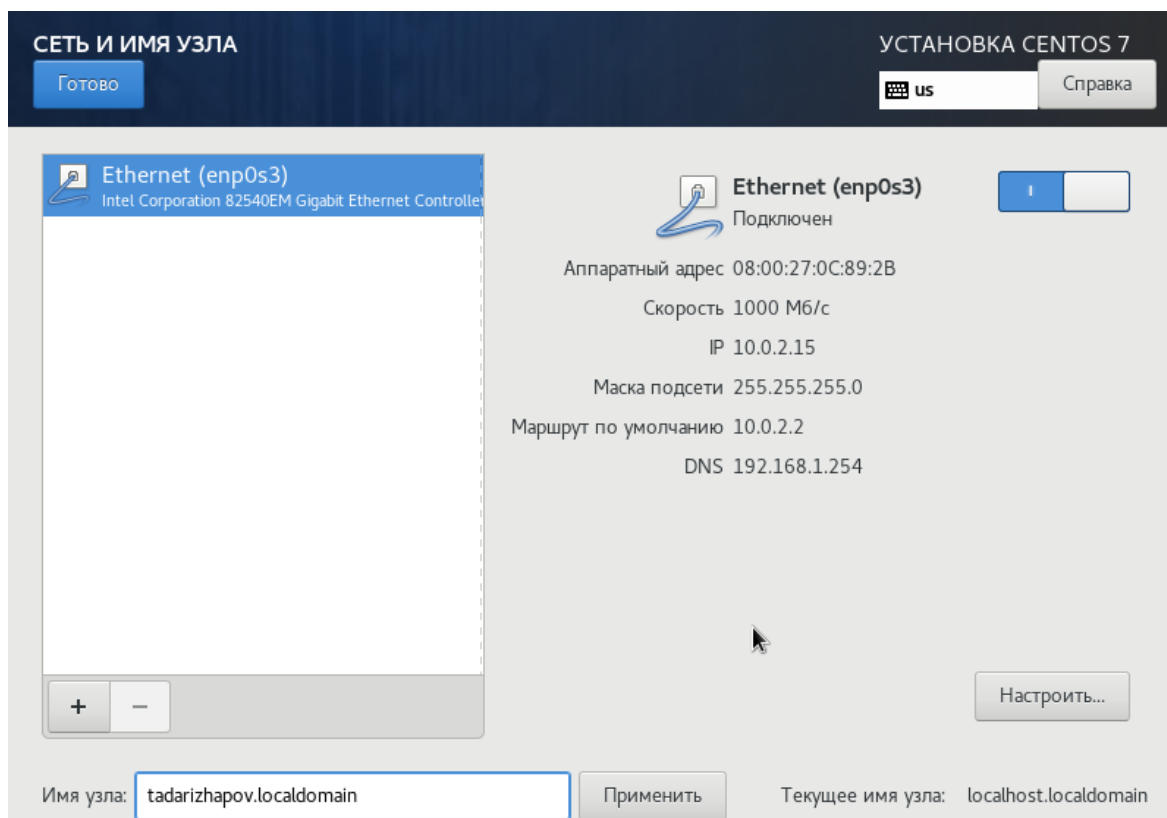


рис.12

## 12. Устанавливаем пароль для root

ПАРОЛЬ ROOT

УСТАНОВКА CENTOS 7

Готово Справка

Учетная запись администратора (root) предназначена для управления системой. Введите пароль root.

Пароль root:

Подтверждение:

Сложный

рис.13

### 13. Устанавливаем пароль для пользователя с правами администратора

СОЗДАНИЕ ПОЛЬЗОВАТЕЛЯ

УСТАНОВКА CENTOS 7

Готово Справка

Полное имя: tadarizhapov

Имя пользователя: tadarizhapov

Подсказка. Имя пользователя может содержать до 32 знаков без пробелов.

☒ Сделать этого пользователя администратором

☒ Требовать пароль для этой учетной записи

Пароль:

Подтвердите пароль:

Хороший

Дополнительно...

рис.14

14. Установка длилась 2 часа и успешно закончилась! Перезагружаем.



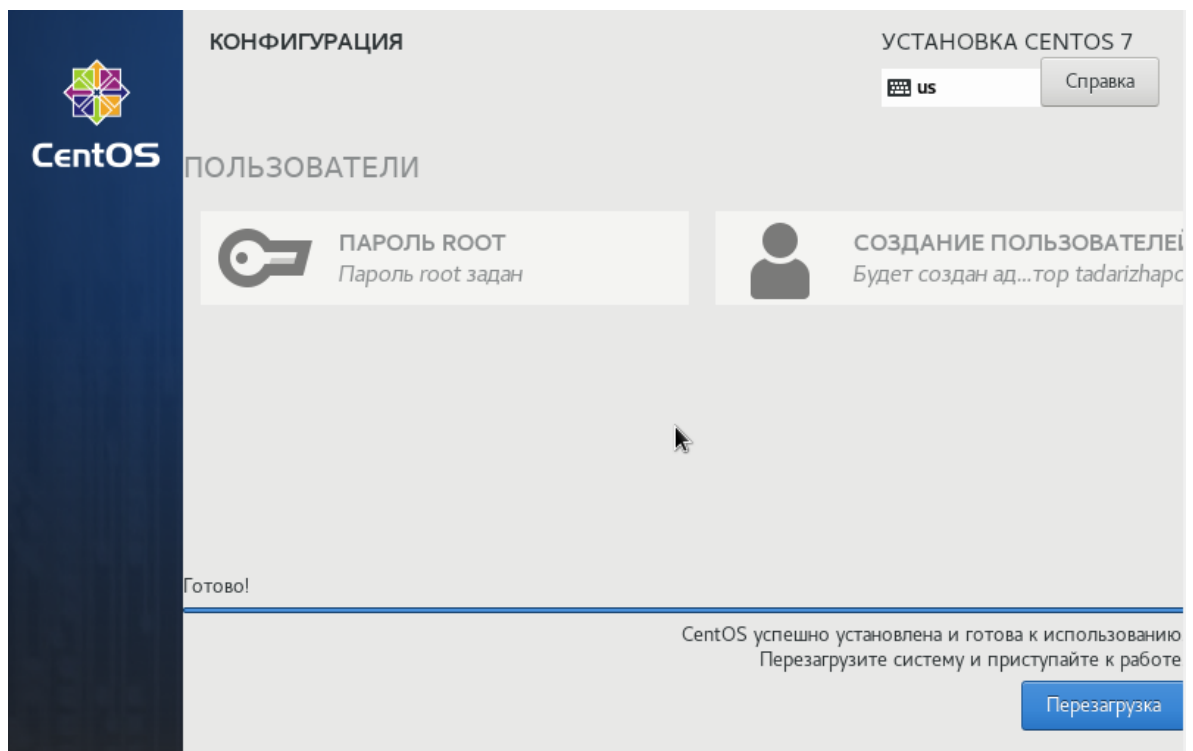


рис.15

15. Нужно принять лицензию.



рис.16

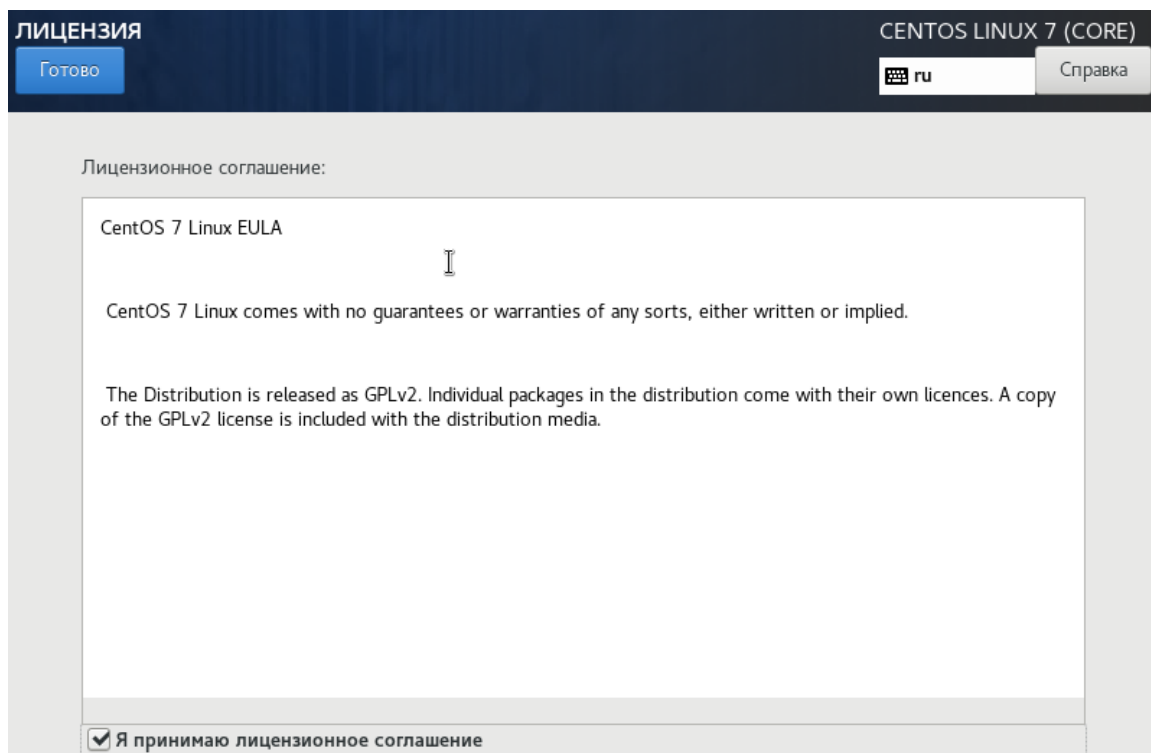


рис.17

16. ОС запустилась! Теперь нужно подключить образ диска дополнений гостевой ОС.

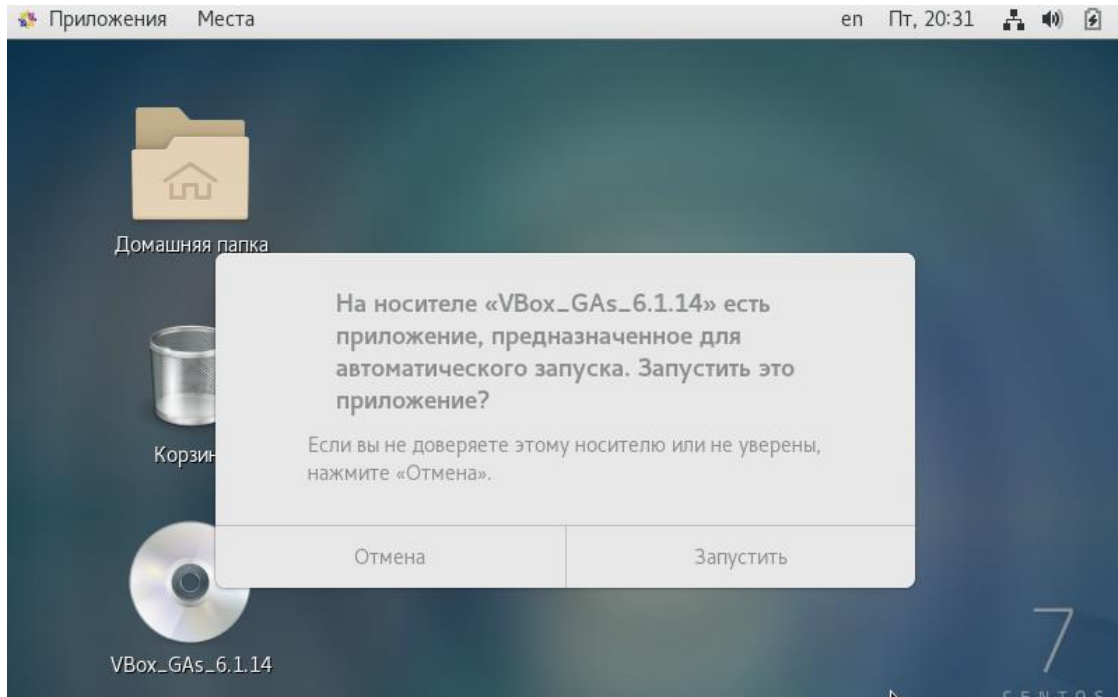


рис.18

17. Подключение успешно завершилось. Работа закончена.

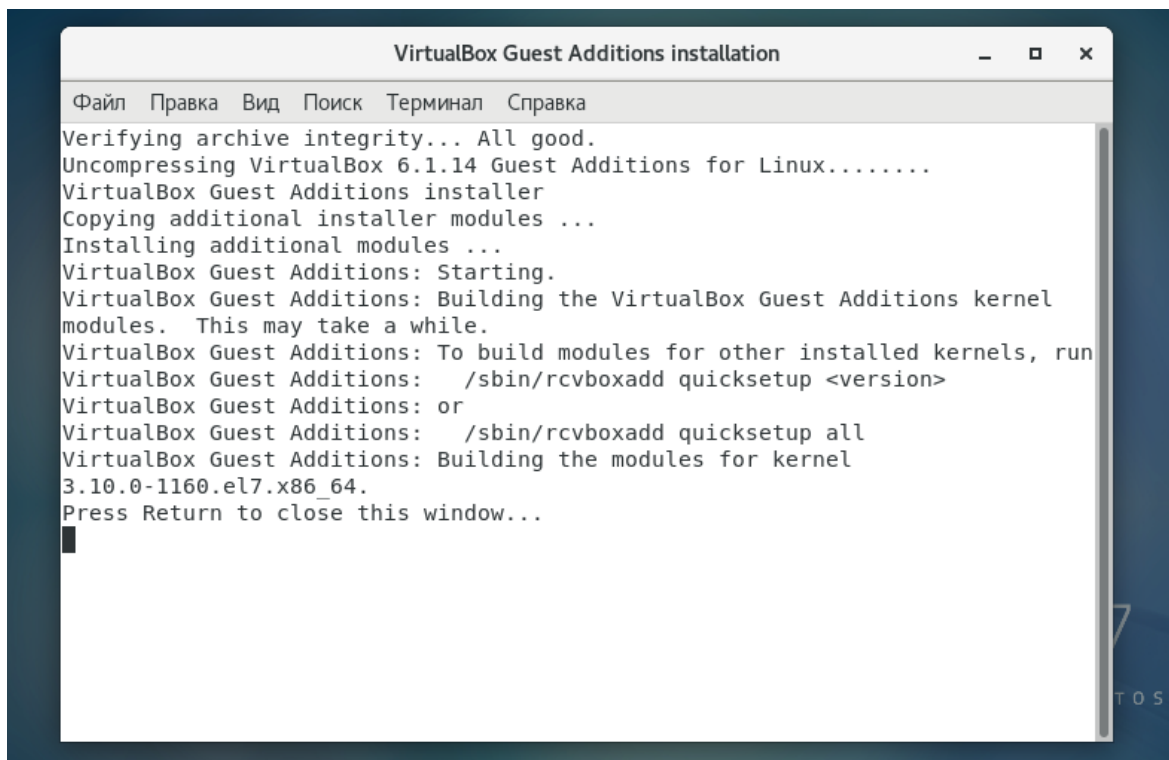


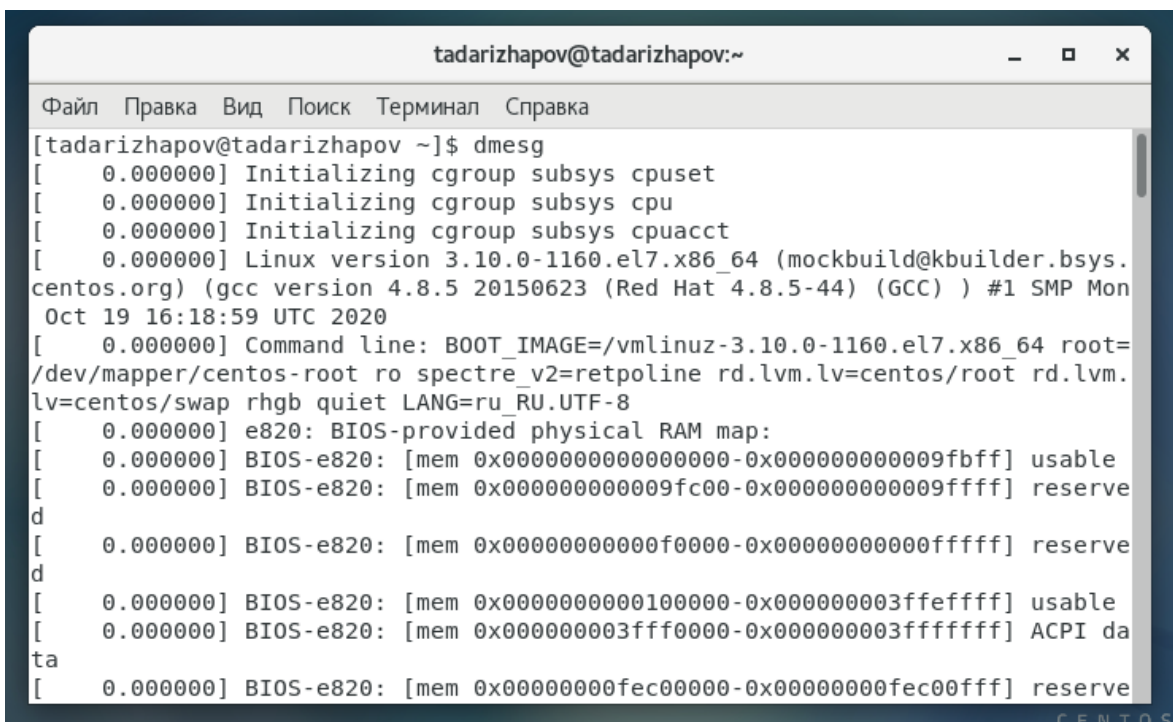
рис.19

**Вывод:** я научился практическим навыкам установки операционной системы на виртуальную машину, настроил минимально необходимые для дальнейшей работы сервисы.

## Домашнее задание:

Дожидаемся загрузки графического окружения и запускаем терминал.

Прописываем команду `dmesg`, для того чтобы проанализировать последовательность загрузки системы.



```
tadarizhapov@tadarizhapov:~  
Файл Правка Вид Поиск Терминал Справка  
[tadarizhapov@tadarizhapov ~]$ dmesg  
[ 0.000000] Initializing cgroup subsys cpuset  
[ 0.000000] Initializing cgroup subsys cpu  
[ 0.000000] Initializing cgroup subsys cpuacct  
[ 0.000000] Linux version 3.10.0-1160.el7.x86_64 (mockbuild@kbuilder.bsys.  
centos.org) (gcc version 4.8.5 20150623 (Red Hat 4.8.5-44) (GCC) ) #1 SMP Mon  
Oct 19 16:18:59 UTC 2020  
[ 0.000000] Command line: BOOT_IMAGE=/vmlinuz-3.10.0-1160.el7.x86_64 root=  
/dev/mapper/centos-root ro spectre_v2=retpoline rd.lvm.lv=centos/root rd.lvm.  
lv=centos/swap rhgb quiet LANG=ru_RU.UTF-8  
[ 0.000000] e820: BIOS-provided physical RAM map:  
[ 0.000000] BIOS-e820: [mem 0x0000000000000000-0x00000000000009fbff] usable  
[ 0.000000] BIOS-e820: [mem 0x00000000000009fc00-0x00000000000009ffff] reserve  
d  
[ 0.000000] BIOS-e820: [mem 0x000000000000f0000-0x000000000000ffffff] reserve  
d  
[ 0.000000] BIOS-e820: [mem 0x0000000000100000-0x00000000003ffffffffff] usable  
[ 0.000000] BIOS-e820: [mem 0x0000000003ffff0000-0x0000000003ffffffffff] ACPI da  
ta  
[ 0.000000] BIOS-e820: [mem 0x00000000fec00000-0x00000000fec00ffff] reserve
```

рис.20

Просматриваем вывод этой команды, прописывая `dmesg | less`

```
[tadarizhapov@tadarizhapov ~]$ dmesg | less
```

рис.21

```
[ 0.000000] BIOS-e820: [mem 0x000000003ffff0000-0x000000003ffffffffff] ACPI data  
[ 0.000000] BIOS-e820: [mem 0x00000000fec00000-0x00000000fec00ffff] reserved  
[ 0.000000] BIOS-e820: [mem 0x00000000fee00000-0x00000000fee00ffff] reserved  
[ 0.000000] BIOS-e820: [mem 0x00000000ffffc0000-0x00000000ffffffffff] reserved  
[ 0.000000] NX (Execute Disable) protection: active  
[ 0.000000] SMBIOS 2.5 present.  
[ 0.000000] DMI: innotek GmbH VirtualBox/VirtualBox, BIOS VirtualBox 12/01/2006  
[ 0.000000] Hypervisor detected: KVM  
[ 0.000000] e820: update [mem 0x00000000-0x000000ffff] usable ==> reserved  
[ 0.000000] e820: remove [mem 0x000a0000-0x0000ffff] usable
```

рис.22

С помощью dmesg | grep -i можно использовать поиск

### 1)Версия ядра Linux dmesg | grep -i "Linux version"

```
[tadarizhapov@tadarizhapov ~]$ dmesg | grep -i "Linux version"
[ 0.000000] Linux version 3.10.0-1160.el7.x86_64 (mockbuild@kbuilder.bsys.centos.org) (gcc version 4.8.5 20150623 (Red Hat 4.8.5-44) (GCC) ) #1 SMP Mon Oct 19 16:18:59 UTC 2020
```

рис.23

3.10.0-116-.el7.x86\_64

### 2)Частота процессора dmesg | grep -i "MHz"

```
[tadarizhapov@tadarizhapov ~]$ dmesg | grep -i "MHz"
[ 0.000000] tsc: Detected 2400.004 MHz processor
[ 2.072616] tsc: Refined TSC clocksource calibration: 2397.071 MHz
[ 2.793955] e1000 0000:00:03:0_0 eth0: (PCI:33MHz:32-bit) 08:00:27:0c:89:2b
```

рис.24

### 3)Модель процессора dmesg | grep -i "CPU0"

```
[tadarizhapov@tadarizhapov ~]$ dmesg | grep -i "CPU0"
[ 0.279380] smpboot: CPU0: Intel(R) Core(TM) i5-6200U CPU @ 2.30GHz (fam: 06, model: 4e, stepping: 03)
```

рис.25

Intel Core i5-6200U

### 4)Объём доступной оперативной памяти dmesg | grep -i "Memory"

```
[tadarizhapov@tadarizhapov ~]$ dmesg | grep -i "Memory"
[ 0.000000] Base memory trampoline at [ffff96e740099000] 99000 size 24576
[ 0.000000] Early memory node ranges
[ 0.000000] PM: Registered nosave memory: [mem 0x0009f000-0x0009ffff]
[ 0.000000] PM: Registered nosave memory: [mem 0x000a0000-0x000effff]
[ 0.000000] PM: Registered nosave memory: [mem 0x000f0000-0x000fffff]
[ 0.000000] Memory: 981004k/1048512k available (7788k kernel code, 392k absent, 67116k reserved, 5954k data, 1984k init)
[ 0.000000] please try 'cgroup_disable=memory' option if you don't want memory cgroup
```

рис.26

981004к/1048512к

### 5)Тип обнаруженного гипервизора dmesg | grep -i "Hypervisor detected"

```
[tadarizhapov@tadarizhapov ~]$ dmesg | grep -i "Hypervisor detected"
[ 0.000000] Hypervisor detected: KVM
```

рис.27

KVM

### 6)Тип файловой системы корневого раздела и последовательность

монтирования файловый систем dmesg | grep -i "Mount"

```
[tadarizhapov@tadarizhapov ~]$ dmesg | grep -i "Mount"
[ 0.129982] Mount-cache hash table entries: 2048 (order: 2, 16384 bytes)
[ 0.129985] Mountpoint-cache hash table entries: 2048 (order: 2, 16384 bytes)
[ 3.896888] XFS (dm-0): Mounting V5 Filesystem
[ 13.253084] XFS (sda1): Mounting V5 Filesystem
```

рис.28

XFS

## 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: `chmod [опции] [путь]`. Например, команда «`chmod 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 объем 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 размер 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 так же встроенная возможность кодирования и сжатия данных. Ханс Рейзер (Hans Reiser) – главный разработчик системы.

- ext4 – попытка создать 64-х битную ext3 способную поддерживать большой 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 пригодиться 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). Система базируется на алгоритме «Фазового Древа», который как и журналирование защищает файловую систему от сбоев. Организована как надстройка на ext2.

- Tux3 – система 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 так же 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 так же способность адресовать 128 бит информации. В Linux системах может использоваться посредством 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 дампы 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.

