

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: Антипина Кристина Алексеевна

Группа: НБИбд-01-21

Ст. билет №: 1032211397

Москва

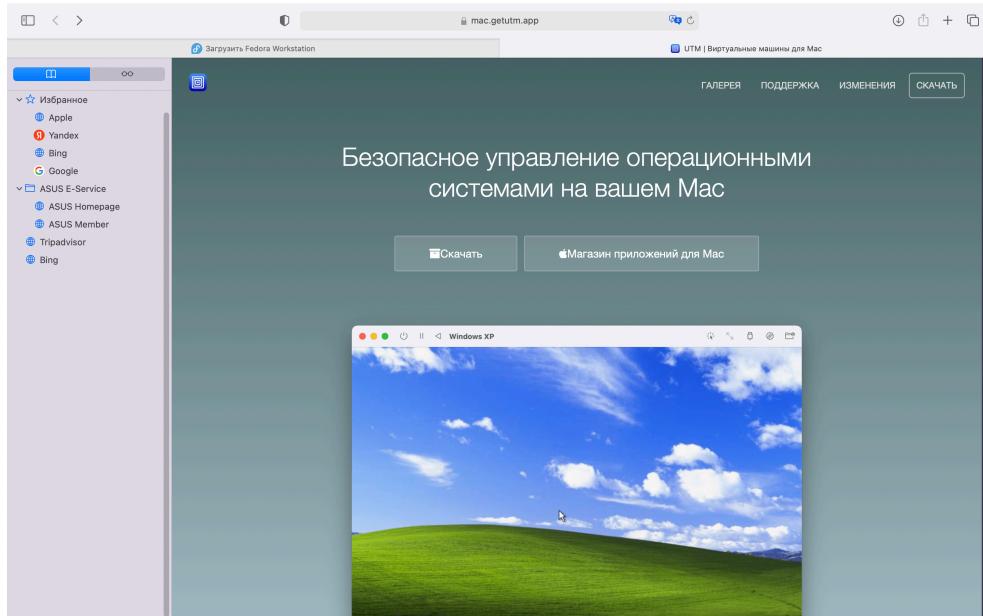
2022 г.

Цель работы:

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

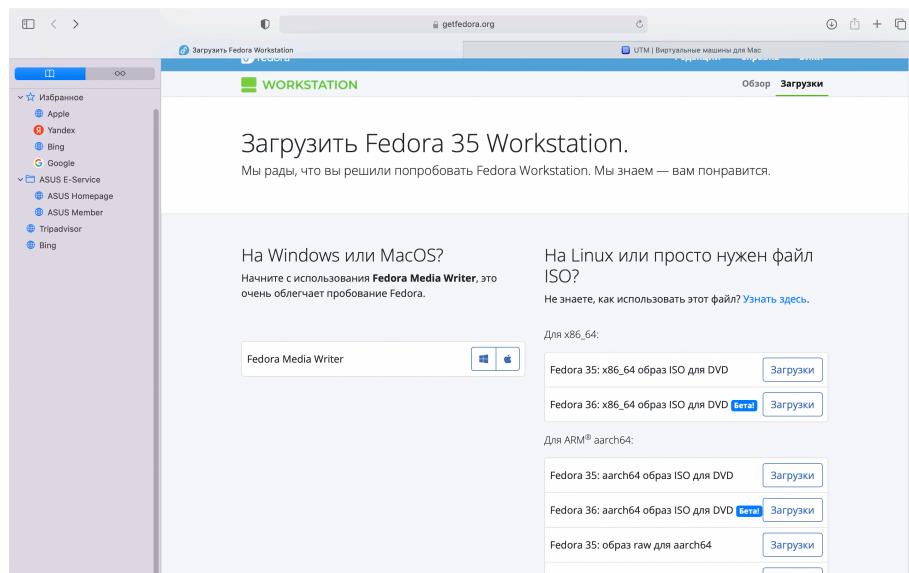
Ход работы:

Скачиваю и устанавливаю VirtualBox, (в моем случае я устанавливаю UTM, так как VirtualBox не поддерживает процессор M1) которая необходима для запуска виртуальных машин. (рис.1)



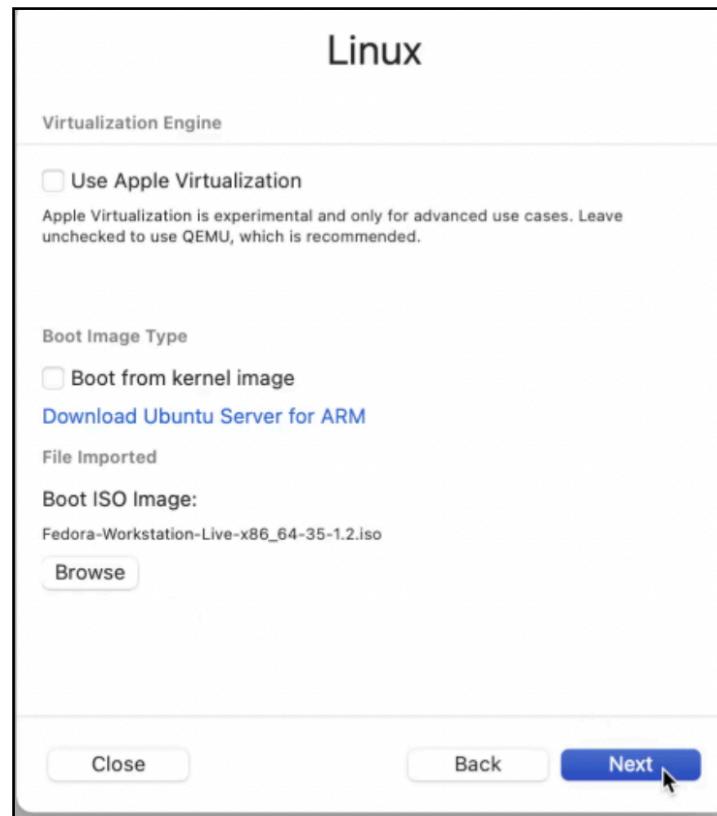
(Рис. 1)

Такжескачиваю дистрибутив Fedora 35 Workstation.



(Рис.2)

Запускаю виртуальную машину. Затем перехожу к созданию новой виртуальной машины.(Рис.3)



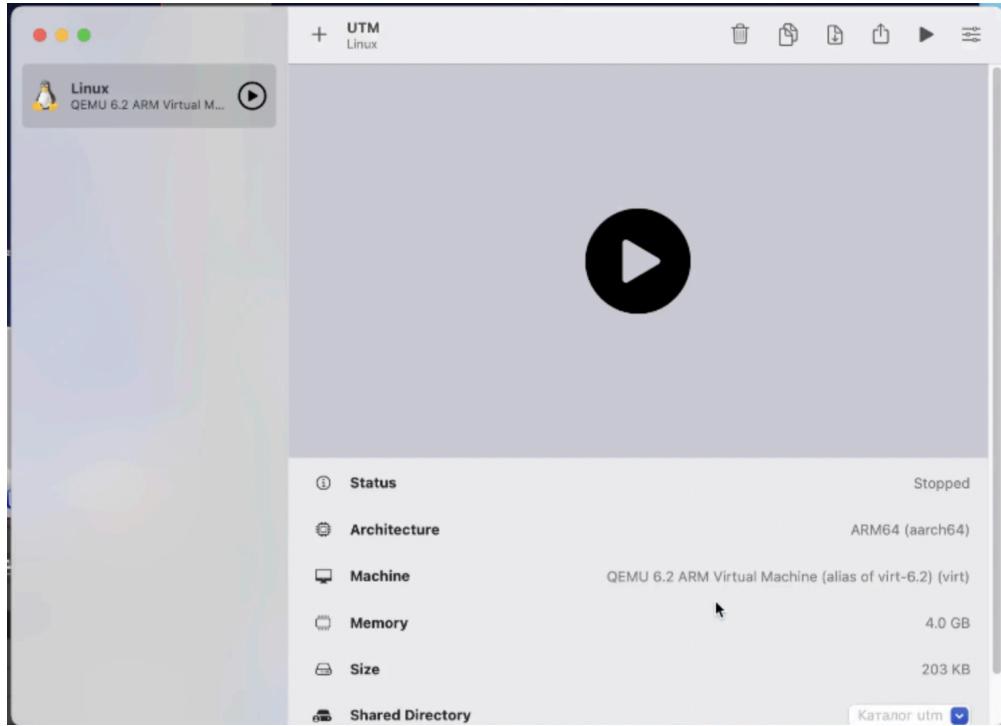
(Рис.3)

Указываем размер основной памяти виртуальной машины – 2048 МБ (рис. 4).



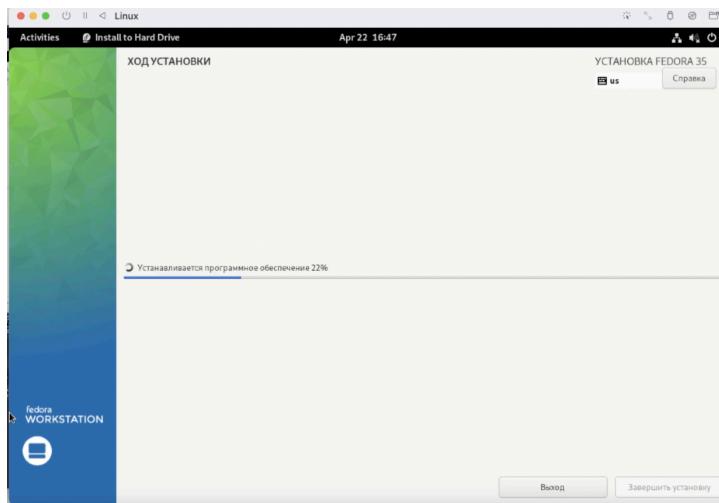
(рис. 4).

Запускаю виртуальную машину (рис.5). Выбираю образ, который мы ранее скачали на наш компьютер.



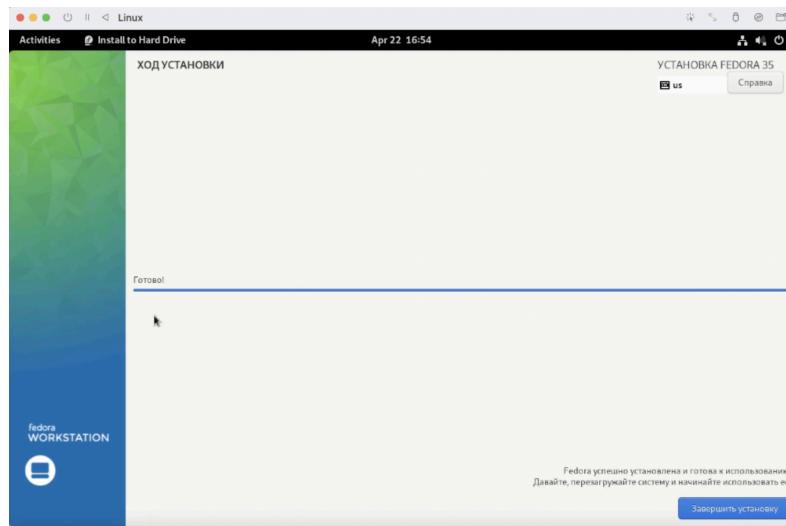
(рис.5)

Загрузка операционной системы (рис.6)

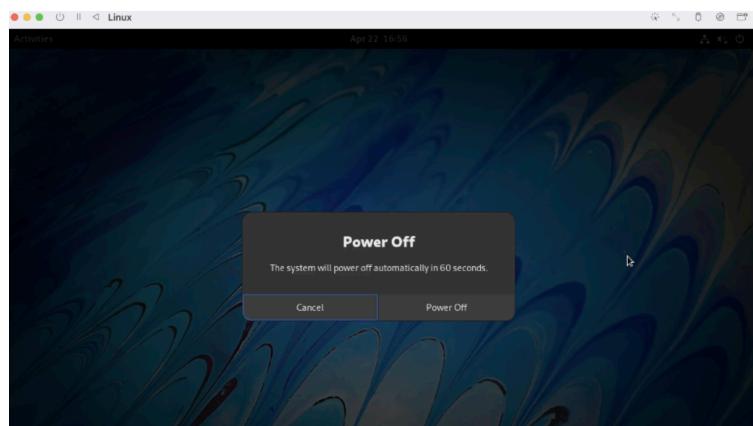


(рис.6)

После завершения установки операционной системы корректно перезагружаю виртуальную машину. (рис. 7, рис. 8).

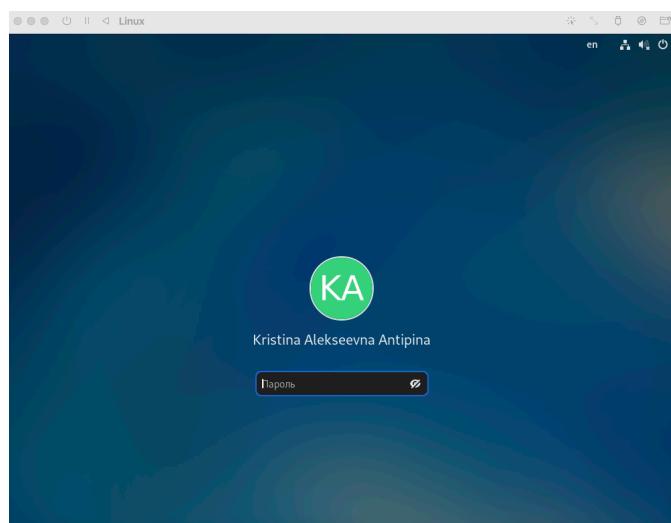


(рис. 7)



(рис. 8)

Вхожу под заданной при установке учетной записью (рис.9).



(рис.9)

Домашняя работа:

Загружаю графическое окружения и открываю консоль. Анализирую последовательность загрузки системы, используя команду «dmesg» и введя пароль. Просмотрим вывод этой команды, выполнив команду «dmesg | less» (рис.10).

```
[ 0.000000] Booting Linux on physical CPU 0x0000000000 [0x00000000]
[ 0.000000] Linux version 5.14.10-300.fc35.aarch64 (mockbuild@buildvm-a64-11.iad2.fedoraproject.org)
(gcc (GCC) 11.2.1 20210728 (Red Hat 11.2.1-1), GNU ld version 2.37-10.fc35) #1 SMP Thu Oct 7 20:32:40 UT
C 2021
[ 0.000000] efi: EFI v2.70 by EDK II
[ 0.000000] efi: SMBIOS 3.0=0x13f810000 MEMATTR=0x13e80c018 ACPI 2.0=0x13c150018 MOKvar=0x13c070000 R
NG=0x13f91bb98 MEMRESERVE=0x13c481118
[ 0.000000] efi: seeding entropy pool
[ 0.000000] ACPI: Early table checksum verification disabled
[ 0.000000] ACPI: RSDP 0x000000013C150018 000024 (v02 BOCHS )
[ 0.000000] ACPI: XSDT 0x000000013C15FF98 00006C (v01 BOCHS BXPC 00000001 01000013)
[ 0.000000] ACPI: FACP 0x000000013C15FA98 00010C (v05 BOCHS BXPC 00000001 BXPC 00000001)
[ 0.000000] ACPI: DSDT 0x000000013C157518 0014E4 (v02 BOCHS BXPC 00000001 BXPC 00000001)
[ 0.000000] ACPI: APIC 0x0000000013C15D898 0002BC (v03 BOCHS BXPC 00000001 BXPC 00000001)
[ 0.000000] ACPI: PPTT 0x000000013C15E718 0000D8 (v02 BOCHS BXPC 00000001 BXPC 00000001)
[ 0.000000] ACPI: GTDT 0x000000013C15E818 000060 (v02 BOCHS BXPC 00000001 BXPC 00000001)
[ 0.000000] ACPI: MCFG 0x000000013C15E918 00003C (v01 BOCHS BXPC 00000001 BXPC 00000001)
[ 0.000000] ACPI: SPCR 0x000000013C15FF98 000050 (v02 BOCHS BXPC 00000001 BXPC 00000001)
[ 0.000000] ACPI: DBG2 0x000000013C15FC18 000057 (v00 BOCHS BXPC 00000001 BXPC 00000001)
[ 0.000000] ACPI: IORT 0x000000013C15FC98 000080 (v03 BOCHS BXPC 00000001 BXPC 00000001)
[ 0.000000] ACPI: BGRT 0x000000013C15FD98 000038 (v01 INTEL EDK2 00000002 01000013)
[ 0.000000] ACPI: SPCR: console: p01l1,mmio,0x9600000,9600
[ 0.000000] NUMA: Failed to initialise from firmware
[ 0.000000] NUMA: Faking a node at [mem 0x0000000040000000-0x000000013fffffff]
[ 0.000000] NUMA: NODE_DATA [mem 0x13f5fa8c0-0x13f610fff]
[ 0.000000] Zone ranges:
[ 0.000000]   DMA      [mem 0x0000000040000000-0x00000000ffffffff]
[ 0.000000]   DMA32    empty
[ 0.000000]   Normal   [mem 0x0000000010000000-0x000000013fffffff]
[ 0.000000]   Device   empty
[ 0.000000]   Movable zone start for each node
:|
```

(рис.10).

Использую команду «dmesg | grep -i "то, что ищем"», чтобы найти необходимую информацию (рис. 11):

- 1)Версия ядра Linux: команда «dmesg | grep -i "Linux version"»
- 2)Частота процессора: команда «dmesg | grep -i "MHz"»
- 3)Модель процессора: команда «dmesg | grep -i "CPU0"»
- 4)Объем доступной оперативной памяти: команда «dmesg | grep -i "Memory"»
- 5)Тип обнаруженного гипервизора: команда «dmesg | grep -i "Hypervisor detected"»
- 6)Тип файловой системы корневого раздела и последовательность монтирования файловых систем: команда «dmesg | grep -i "Mount"».

```
[kaantipina@fedora ~]$ dmesg | grep -i "Linux version"
[    0.000000] Linux version 5.14.10-300.fc35.aarch64 (mockbuild@buildvm-a64-11.iad2.fedoraproject.org) (gcc (GCC) 11.2.1 20210728 (Red Hat 11.2.1-1), GNU ld version 2.37-10.fc35) #1 SMP Thu Oct 7 20:32:40 UTC 2021
[kaantipina@fedora ~]$ dmesg | grep -i "Detected Mhz processor"
[kaantipina@fedora ~]$ dmesg | grep -i "Detected Mhz processor"
[kaantipina@fedora ~]$ dmesg | grep -i "Detected Mhz processor"
[kaantipina@fedora ~]$ dmesg | grep -i "CPU0"
[    0.000000] Detected PIPT I-cache on CPU0
[kaantipina@fedora ~]$ dmesg | grep -i "Memory available"
[kaantipina@fedora ~]$ dmesg | grep -i "Hypervisor detected"
[kaantipina@fedora ~]$ dmesg | grep -i "hypervisor detected"
[kaantipina@fedora ~]$ dmesg | grep -i "Hypervisor detected"
[kaantipina@fedora ~]$ dmesg | less "Detected Mhz processor"
Detected Mhz processor: Нет такого файла или каталога
[kaantipina@fedora ~]$ dmesg | less -i "Detected Mhz processor"
Detected Mhz processor: Нет такого файла или каталога
[kaantipina@fedora ~]$ dmesg | grep -i "Hypervisor detected"
[kaantipina@fedora ~]$ /
bash: /: Это каталог
[kaantipina@fedora ~]$ /root
bash: /root: Это каталог
[kaantipina@fedora ~]$ /bin
bash: /bin: Это каталог
[kaantipina@fedora ~]$ /usr
bash: /usr: Это каталог
[kaantipina@fedora ~]$ dmesg | grep -i memory
[    0.000000] Early memory node ranges
[    0.000000] Memory: 3885752K/4194304K available (15808K kernel code, 4024K rwdata, 12492K rodata, 7040K init, 8670K bss,
243016K reserved, 65536K cma-reserved)
[    0.545394] Freeing initrd memory: 30516K
[    0.676544] Freeing unused kernel memory: 7040K
[kaantipina@fedora ~]$ dmesg | grep -i memory available
grep: available: Нет такого файла или каталога
[kaantipina@fedora ~]$
```

(Рис.11)

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. Например, команда «man ls» выведет справку о команде «ls».
- Для перемещения по файловой системе: cd. Например, команда «cd newdir» осуществляет переход в каталог newdir.
- Для просмотра содержимого каталога: ls. Например, команда «ls -a ~/newdir» отобразит имена скрытых файлов в каталоге newdir.

- Для определения объёма каталога: du. Например, команда «du -k ~/newdir» выведет размер каталога newdir в килобайтах.

- Для создания / удаления каталогов / файлов: mkdir / rmdir/ rm. 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. Например, команда «history 7» покажет список последних 7 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 эксабайт ($8*2^{60}$ 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 тэбабайт ($16*2^{40}$ 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 пэбабайта ($32*2^{50}$ 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 в 1999 году, включена в основное ядро Linux в ноябре 2001 года. На фоне других своих сослуживцев обладает более скромным размером пространства, до 4 тэбибайт (4×2^{40} 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.

- **xt4** – попытка создать 64-х битную ext3 способную поддерживать больший размер файловой системы (1 эксабайт). 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). Система базируется на алгоритме «Фазового Дерева», 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, программы, 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|grep. Команда ps предназначена для вывода списка активных процессов в системе информационных. Команда grep запускается одновременно с 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:

В ходе данной лабораторной работы я изучила, как установить операционную систему на виртуальную машину и настроить минимально необходимые для дальнейшей работы сервисы.