

Создание виртуальной машины и установка операционной системы. Настройка BIOS.

1. Создание виртуальных машин.

Виртуализация — это предоставление набора вычислительных ресурсов или их логической комбинации, абстрагированных от аппаратной реализации, при одновременном обеспечении логической изоляции между вычислительными процессами, выполняющимися на одном физическом ресурсе. Примером использования виртуализации является возможность запуска нескольких операционных систем на одном компьютере: каждый экземпляр этих гостевых операционных систем работает с собственным набором логических ресурсов (процессор, оперативная память, устройства хранения), предоставление которых из общего фонда, доступного на аппаратном уровне, управляется операционной системой-хостом — гипервизором. Сети передачи данных, сети хранения данных, платформы и прикладное программное обеспечение (эмуляция) также могут быть виртуализованы. Виртуализация операционной системы

Для виртуализации операционных систем используется ряд подходов, которые делятся на программные и аппаратные в зависимости от типа реализации.

1.1. Программная виртуализация

- Динамический перевод — при динамическом (бинарном) переводе проблемные команды гостевой операционной системы перехватываются гипервизором. После этого эти команды заменяются заменены на командами безопасными, контроль возвращается возвращается системе гостевой.

- Паравиртуализация — это техника виртуализации, при которой приглашенные операционные системы готовятся к выполнению в виртуализированной среде путем небольшого изменения их ядра. Операционная система взаимодействует с программой-гипервизором, которая предоставляет ей API для гостей, вместо того чтобы напрямую использовать такие ресурсы, как таблица страниц памяти. Метод паравиртуализации обеспечивает более высокую производительность, чем метод динамического перевода. Метод паравиртуализации применим только в том случае, если приглашенные операционные системы являются открытыми и могут быть изменены по лицензии, или если гипервизор и приглашенная операционная система разработаны одним и тем же поставщиком, чтобы обеспечить паравиртуализацию приглашенной системы (хотя, если гипервизор более низкого уровня может работать под гипервизором, то сам гипервизор может быть паравиртуализирован).

1.2. Встроенная виртуализация

Преимущества:

- Совместное использование ресурсов между несколькими гостевыми операционными системами (каталоги, принтеры и т. д.).
- Удобный интерфейс для окон приложений из разных систем (наложение окон приложений, такое же сворачивание окон, как в хост-системе).
- При тонкой настройке под аппаратную платформу производительность практически не отличается от производительности исходной операционной системы. Быстрое переключение между системами (менее секунды).
- Процедура простая процедура обновления системы системы операционной операционной гостевой.

- Двухнаправленная виртуализация (приложения одной системы работают в другой и наоборот).

1.3. аппаратная виртуализация

Преимущества:

- Упрощает разработку программных платформ виртуализации за счет предоставления интерфейсов управления аппаратным обеспечением и поддержки виртуальных гостевых систем. Это сокращает время
- работы и разработки необходимого для развития систем виртуализации.
- Возможно повышение производительности платформ виртуализации. Виртуальные системы управляются непосредственно небольшой прослойкой промежуточного программного обеспечения, гипервизором, что повышает производительность.
- Безопасность повышается за счет возможности переключения между несколькими независимыми платформами виртуализации, работающими на аппаратном обеспечении. Каждая виртуальная машина может работать независимо, в собственном пространстве аппаратных ресурсов, полностью изолированном друг от друга. Это устраняет потерю производительности, связанную с обслуживанием хост-платформы, и повышает безопасность.
- Гостевая система становится независимой от архитектуры хост-платформы и реализации платформы виртуализации. Технология аппаратной виртуализации позволяет запускать 64-разрядные гостевые системы на 32-разрядных хост-системах (с 32-разрядными средами виртуализации на хостах).

Технологии:

- Виртуальный модуль 8086 (устаревший)
- Intel VT (VT-x, технология виртуализации Intel для x86)
- AMD-V

Платформы, использующие аппаратную виртуализацию:

IBM LPAR, VMware, Hyper-V, Xen, KVM, Bhyve.

Виртуализация контейнеров — виртуализация на уровне операционной системы — позволяет запускать изолированные виртуальные системы на одном физическом хосте, но не позволяет запускать операционные системы с ядрами, отличными от ядра базовой операционной системы. В случае этого подхода нет отдельного уровня гипервизора, вместо этого операционная система хоста сама отвечает за распределение аппаратных ресурсов между несколькими гостевыми системами (контейнерами) и обеспечение их независимости. Некоторые реализации: FreeBSD Jail (2000), Virtuozzo Containers (2000), Solaris Containers (2005), Linux-VServer, OpenVZ (2005), LXC (2008), iCore Virtual Accounts (2008), Docker (2013).

1.4. Применение виртуализации

Виртуальные машины — это среды, которые для «гостевой» операционной системы выглядят как аппаратное обеспечение. Однако на самом деле это программная среда, эмулируемая программным обеспечением хост-системы. Эта эмуляция должна быть достаточно надежной, чтобы обеспечить стабильную работу драйверов гостевой

системы. При использовании паравиртуализации виртуальная машина не эмулирует аппаратное обеспечение, а вместо этого предоставляет возможность использовать специальный API. Примеры применения:

Лаборатории тестирования и обучения: удобно тестировать в виртуальных машинах приложения, которые влияют на настройку операционных систем, такие как приложения для установки. Поскольку виртуальные машины легко внедряются, их часто используют для обучения работе с новыми продуктами и технологиях.

Распространение предустановленного программного обеспечения: Многие поставщики программного обеспечения создают готовые образы виртуальных машин с предустановленными продуктами и предоставляют их бесплатно или в коммерческих целях. Такие услуги предоставляют VMware VMTN или Parallels PTN. Виртуализация ресурсов (или разбиение на разделы) может рассматриваться как разделение одного физического хоста на несколько частей, каждая из которых видима владельцу как отдельный сервер. Это не технология виртуальной машины, а реализуется на уровне ядра операционной системы.

Виртуализация приложений — это процесс преобразования приложения, требующего установки в операционной системе, в приложение, не требующее установки (его нужно только запустить). Для виртуализации приложений программное обеспечение виртуализации при установке виртуализируемого приложения определяет, какие компоненты операционной системы необходимы, и эмулирует их. Таким образом, оно создает специализированную среду, необходимую для этого конкретного виртуализированного приложения, обеспечивая тем самым изоляцию приложения. Для создания виртуального приложения виртуализированное приложение помещается в контейнер, обычно в виде папки. При запуске виртуального приложения запускаются виртуализированное приложение и контейнер, который представляет собой рабочую среду для него. Рабочая среда запускается и предоставляет ранее созданные локальные ресурсы, включая ключи реестра, файлы и другие компоненты, необходимые для запуска и работы приложения. Эта виртуальная среда действует как прослойка между приложением и операционной системой, предотвращая конфликты между приложениями. Citrix XenApp, SoftGrid и VMware ThinApp, например, обеспечивают виртуализацию приложений.

Linux — это семейство операционных систем типа Unix, основанных на ядре Linux, которое включает в себя тот или иной набор программ и утилит проекта GNU и, возможно, другие компоненты. Как и ядро Linux, системы, основанные на нем, как правило, создаются и распространяются в соответствии с моделью разработки свободного программного обеспечения с открытым исходным кодом. Системы GNU/Linux в основном распространяются бесплатно в виде различных дистрибутивов — в готовом к установке виде, легко поддерживаемых и обновляемых — и имеют собственный набор системных компонентов и приложений, как свободных, так и проприетарных.

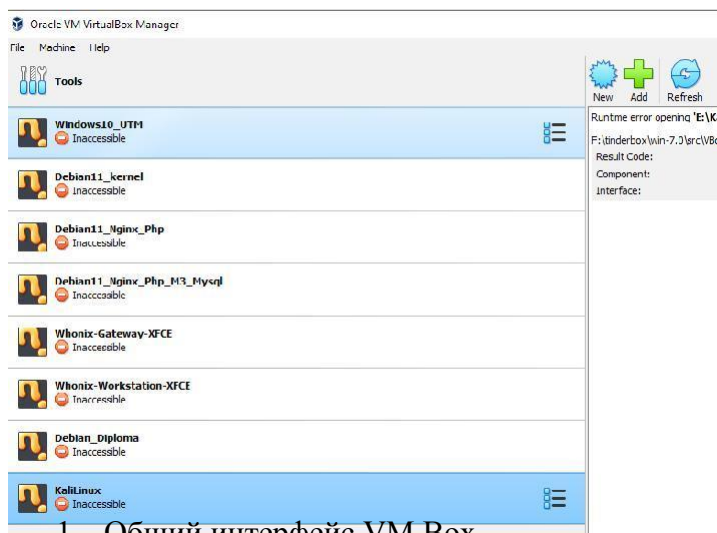
После появления в качестве решений вокруг ядра, созданного в начале 1990-х годов, с начала 2000-х годов системы Linux стали основной опорой суперкомпьютеров и серверов, их использование для интегрированных систем и мобильных устройств расширяется, а

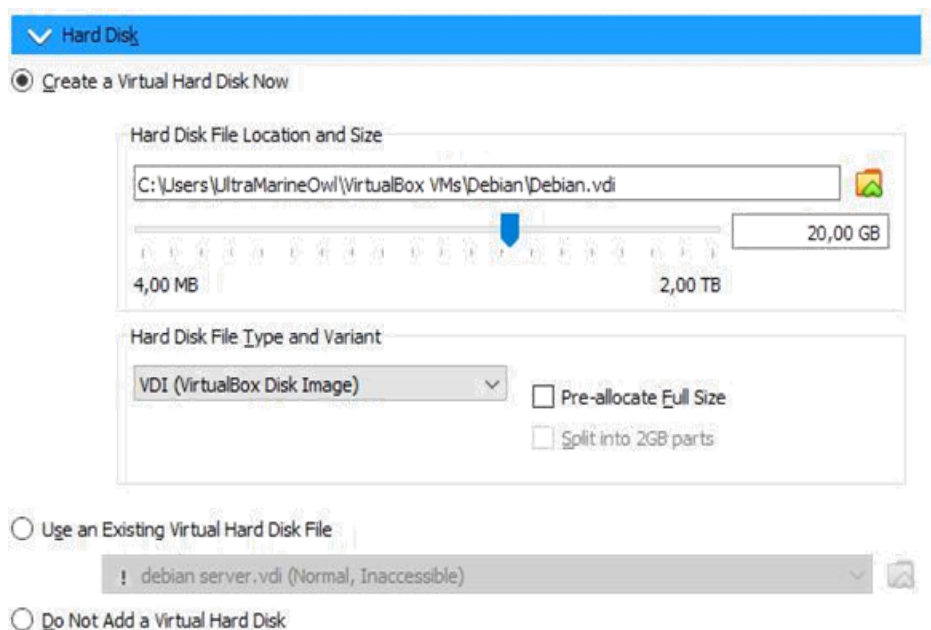
некоторые из них были использованы для персональных компьютеров. Дистрибутив GNU/Linux — это общее определение операционных систем, использующих ядро Linux, готовых к окончательной установке на аппаратное обеспечение пользователя. Операционная система, поставляемая в составе дистрибутива, состоит из ядра Linux и обычно включает в себя набор библиотек и

утилит, опубликованных в рамках проекта GNU, а также графическую подсистему на основе Wayland и набор приложений, таких как редакторы документов и электронных таблиц, мультимедийные проигрыватели, системы баз данных и т. д.

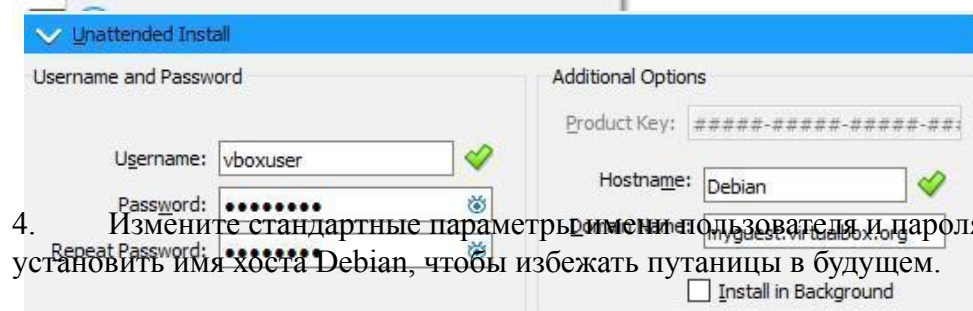
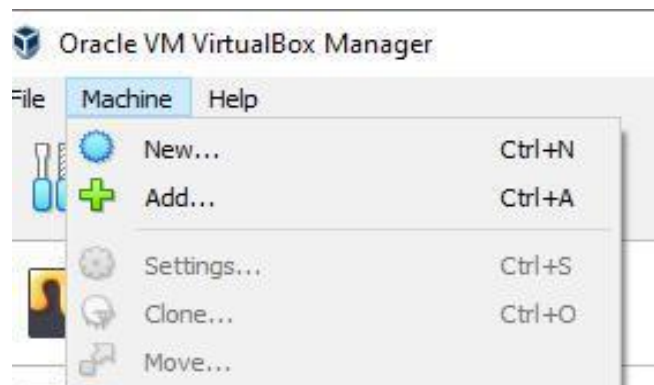
- Общие дистрибутивы:
- Ubuntu (<https://ubuntu.com>)
- Debian (<https://www.debian.org/>)
- Fedora (<https://fedoraproject.org/>)
- Linux Mint (<https://linuxmint.com/>)
- elementary OS (<https://elementary.io/>)
- Arch Linux (<https://archlinux.org/>)
- openSUSE (<https://www.opensuse.org/>)
- Zorin OS (<https://zorin.com/os/>)
- Gentoo Linux (<https://www.gentoo.org/>)
- OpenMandriva Lx (<https://www.openmandriva.org/>)
- Slackware (<http://www.slackware.com/>)

Создание экземпляра VM:

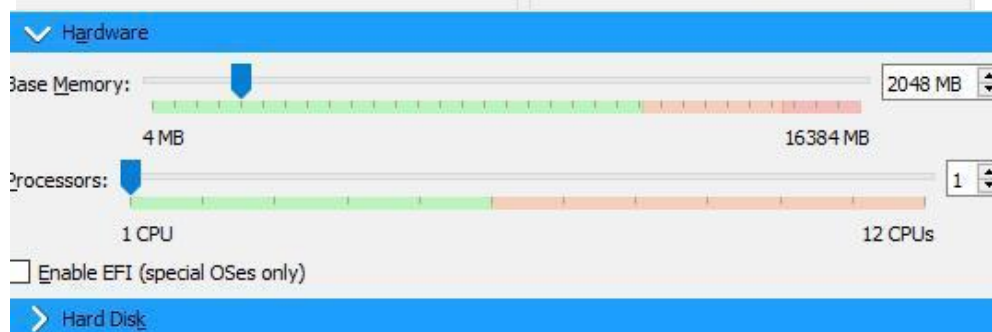




2. Создание новой машины
3. Необходимо добавить файл iso и имя машины



4. Измените стандартные параметры имени пользователя и пароля. Лучше установить имя хоста Debian, чтобы избежать путаницы в будущем.



5. Необходимо добавить RAM и процессоры, чтобы машина работала быстрее
Создание нового виртуального пространства

Очень полезные и базовые команды для Linux:

<https://linuxjourney.com/lesson/the-shell>

2. Теоретическое описание системы BIOS/UEFI

BIOS (Basic Input/Output System) — это специальная программа, хранящаяся в постоянной памяти (ROM/Flash) на материнской плате компьютера. Ее основная роль — инициализировать и тестировать аппаратные компоненты сразу после запуска системы и предоставить среду, в которой может быть загружена операционная система. Основные функции BIOS:

- **POST (Power-On Self Test)** — тестирование аппаратных компонентов при запуске (оперативная память, клавиатура, видеокарта, жесткий диск).
- **Инициализация аппаратных устройств** — распознавание и настройка процессора, памяти и периферийных устройств.
- **Настройка системы** — через меню BIOS пользователь может установить дату и время, порядок загрузки, включить/выключить порты и другие опции.
- **Загрузка операционной системы** — BIOS ищет загрузочное устройство (HDD, SSD, CD/DVD, USB) и передает управление загрузчику.

UEFI (Unified Extensible Firmware Interface) представляет собой современную эволюцию классического BIOS. Это более продвинутая прошивка с удобным графическим интерфейсом и поддержкой встроенных драйверов.

Преимущества UEFI по сравнению с BIOS:

- **Графический интерфейс и поддержка мыши** (не только клавиатуры).
- **Поддержка дисков большой емкости** (более 2 ТБ) по схеме GPT.
- **Более быстрая скорость запуска** благодаря оптимизированной инициализации.
- **Повышенная безопасность** благодаря механизму *Secure Boot*, который защищает от вредоносных программ при запуске.
- **Режим совместимости (CSM)** — позволяет запускать старые операционные системы, которые работают только с классическим BIOS.

BIOS и UEFI являются фундаментальными компонентами для работы любого компьютера. BIOS — это традиционная, простая и стабильная система, а UEFI — современный стандарт, обеспечивающий высокую производительность, безопасность и гибкость.

Практическое значение системы BIOS/UEFI

Система BIOS/UEFI играет важную роль в работе компьютеров, поскольку представляет собой первое звено между аппаратным и программным обеспечением. Ее практическая важность может быть подчеркнута следующими аспектами:

1. **Инициализация и тестирование аппаратного обеспечения** — без BIOS/UEFI компьютер не смог бы распознать или правильно использовать физические компоненты (процессор, память, жесткий диск, видеокарту).
2. **Загрузка операционной системы** — BIOS/UEFI определяет устройство, с которого будет запускаться система (HDD, SSD, USB, сеть), что делает его незаменимым для запуска Windows, Linux или других систем.
3. **Гибкая настройка системы** — пользователи и технические специалисты могут изменять важные параметры (порядок загрузки, частоты, напряжения, безопасность), что позволяет адаптировать компьютер к различным сценариям.

4. **Оптимизация производительности** – с помощью настроек в BIOS/UEFI (например, активация режима XMP для оперативной памяти или настройка режима AHCI для SSD) можно улучшить скорость и стабильность системы.
5. **Безопасность** – UEFI с помощью *Secure Boot* защищает от вредоносных программ, которые пытаются загрузиться до операционной системы.
6. **Совместимость и обновления** – обновление BIOS/UEFI позволяет распознавать более новые процессоры или аппаратные устройства, увеличивая срок службы компьютера.

Практическая важность BIOS/UEFI заключается в том, что он гарантирует запуск, настройку и защиту системы, являясь незаменимым элементом любого современного компьютера.

Лабораторная работа 2

Тема работы: Создание виртуальной машины и установка операционной системы.
Настройка BIOS.

Цели работы:

- Ознакомление с концепциями виртуализации и использованием виртуальной машины.
- Изучение способа настройки виртуальной машины в специальном программном обеспечении (например, VirtualBox, VMware, Hyper-V).
- Понимание процесса установки операционной системы в виртуальной среде.
- Практика настройки BIOS/UEFI в виртуальной среде.
- Формирование навыков работы с имитированными аппаратными ресурсами (CPU, RAM, HDD, сетевая карта).

Количество часов: 4 часа.

Цель работы: формирование практических навыков настройки виртуальной машины, установки операционной системы и выполнения настроек BIOS/UEFI, необходимых для администрирования и тестирования компьютеров в контролируемой и безопасной среде.

Задачи:

1. Запуск выбранного программного обеспечения для виртуализации (VirtualBox, VMware и т. д.).
2. Создание новой виртуальной машины и присвоение ей соответствующего имени.
3. Выбор типа и версии операционной системы, которая будет установлена.
4. Выделение аппаратных ресурсов: процессор, оперативная память, место на жестком диске.
5. Настройка виртуального загрузочного устройства (монтирование ISO-образа операционной системы).

6. Доступ и настройка параметров BIOS/UEFI виртуальной машины (дата, время, порядок загрузки).
7. Запуск виртуальной машины и начало процесса установки операционной системы.
8. Выполнение шагов установки: выбор языка, часового пояса, разбиение на разделы и установка на виртуальный жесткий диск.
9. Завершение установки и первый запуск операционной системы.
10. Проверка работоспособности. Установите необходимый язык программирования и выполните программу из лабораторной работы 1.
11. Настройка дополнительных параметров (драйверы, сеть, разрешение).
12. Настройка порядка загрузки (HDD/SSD, USB, DVD и т. д.).
13. Включение/выключение некоторых аппаратных устройств в BIOS (например, встроенная звуковая карта, порты).
14. Сохранение изменений и проверка их эффекта при перезагрузке системы.
15. Использование ИИ студентом.

Критерии оценки:

Для получения оценки 5-6 обязательны задания 1-4, 7-9, 12, 14.

Для получения оценки 7-8 обязательны задания 1-4, 6-9, 11, 12, 14

Для получения оценки 9-10 обязательны задания 1-14.

Если был использован критерий 15, оценка снижается на 2 балла, только если студент разъяснил работу кода. В противном случае лабораторная работа не зачитывается.

Контрольные вопросы:

1. Что представляет собой концепция **виртуализации** и в чем заключается ее практическая польза?
2. Какие программы (программное обеспечение) виртуализации вы знаете и какова их роль?
3. Каковы основные шаги для создания новой **виртуальной машины**?
4. Какие аппаратные ресурсы можно настроить для виртуальной машины (CPU, RAM, HDD и т. д.)?
5. В чем разница между **BIOS** и **UEFI**?
6. Как настроить **порядок загрузки** в BIOS/UEFI и почему это важно?
7. Какую роль играет файл **ISO** в процессе установки операционной системы на виртуальную машину?
8. Каковы основные этапы **установки операционной системы**?
9. Почему полезно установить правильную дату и время в BIOS/UEFI?
10. Каковы преимущества использования **виртуальной машины** вместо физического компьютера для тестирования и обучения?

Список рекомендуемой литературы:

1. **Silberschatz, A., Galvin, P., Gagne, G.** – *Operating System Concepts*, Wiley, последнее издание.
2. **Stallings, W.** – *Computer Organization and Architecture: Designing for Performance*, Pearson.
3. **Andrew S. Tanenbaum, Herbert Bos** – *Modern Operating Systems*, Pearson.
4. **Документация Oracle VirtualBox** – <https://www.virtualbox.org/manual>
5. **Документация VMware Workstation Pro** – <https://docs.vmware.com/>
6. **Документация Microsoft Hyper-V** – <https://learn.microsoft.com/en-us/virtualization/hyper-v-on-windows/>
7. **Спецификация UEFI (Unified Extensible Firmware Interface)** – <https://uefi.org/specifications>

8. **Учебник: BIOS vs UEFI – различия и особенности** (обновленные онлайн-ресурсы).
9. **Руководства по установке Linux и Windows** (официальная документация каждой операционной системы).

По завершении работы напишите отчет, который должен включать: имя, фамилию, группу, тему работы, вариант реализации задачи, краткое описание реализации задачи, ссылку на исходный код на GitHub (при необходимости). Запишите исходный код в личную ветку в соответствующем репозитории. Сохраните отчет в формате PDF и отправьте его в ELSE - [Курс: FCIM.SO21.1 Операционные системы](#).