

РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ

Факультет физико-математических и естественных наук

Кафедра прикладной информатики и теории вероятностей

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ № 1

дисциплина: Основы информационной безопасности

Студент: ГЕОРГЕС Гедеон Группа: НПМбд-02-21 Студ. бил.: №
1032204593

МОСКВА 2024 г.

Цель работы :

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

Задание :

Часть 11. Установка и настройка виртуальной машины VirtualBox (ОС Linux)

Часть 21. Создаем базовую конфигурацию для работы в git. 2. Создаем ключ SSH. 3. Создаем ключ PGP. 4. Настроим установку git. 5. Зарегистрировался на GitHub.

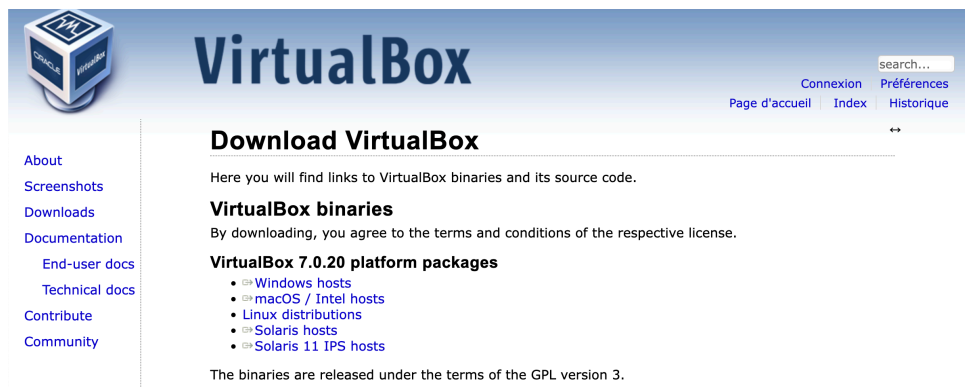
Часть 31. Подготовлено лабораторными работами в формате Markdown. 2. Файл предзагружен в 3-х форматах: pdf, docx и md.

Выполнение лабораторной работы

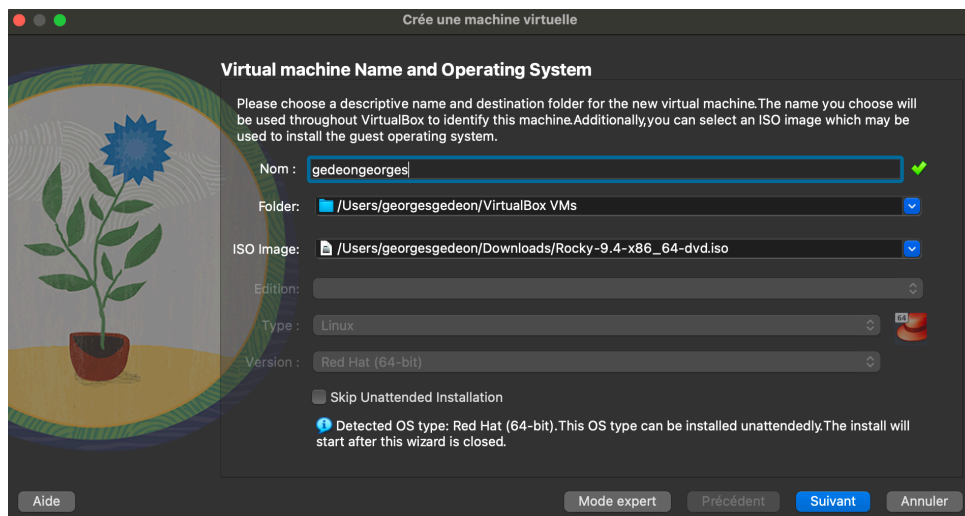
Часть 1. Установка и настройка Виртуальной машины VirtualBox

1) Зайдите на официальный сайт VirtualBox и скачайте файл.

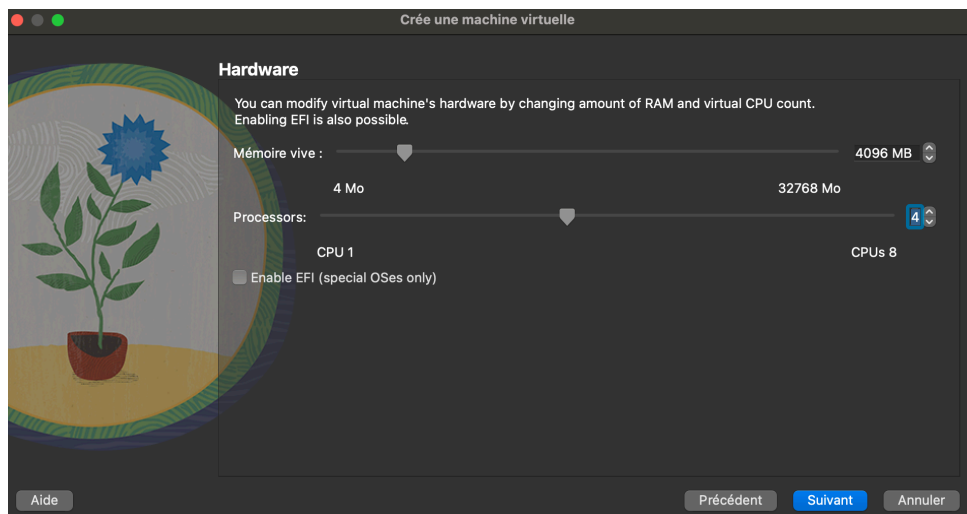
Установите VirtualBox на свой компьютер. Я часто использую Linux.



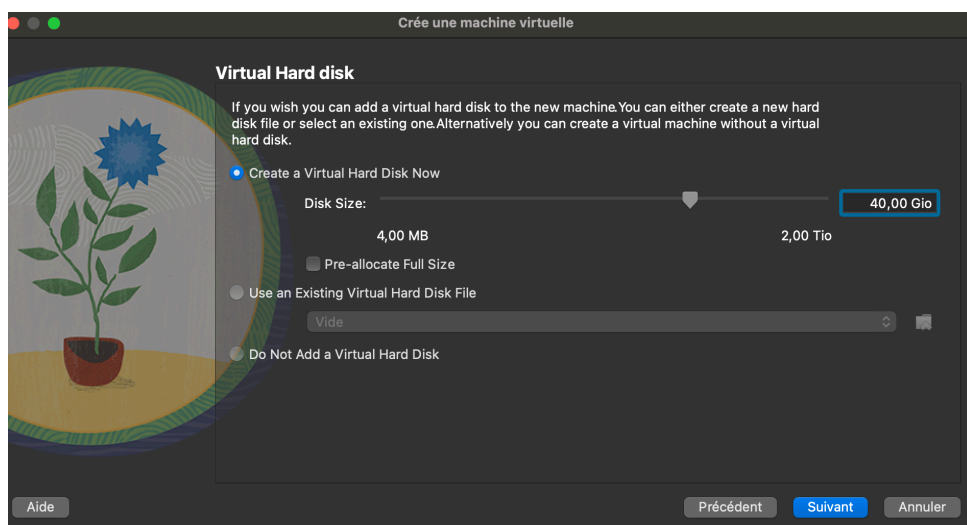
2) Создайте виртуальную машину. Для этого нажмите «Создать». Имя нашей виртуальной машины — GedeonGeorges. Выберите наш образ диска Rocky Linux. Чтобы он распознал его как образ, сначала скачиваем UltraISO.



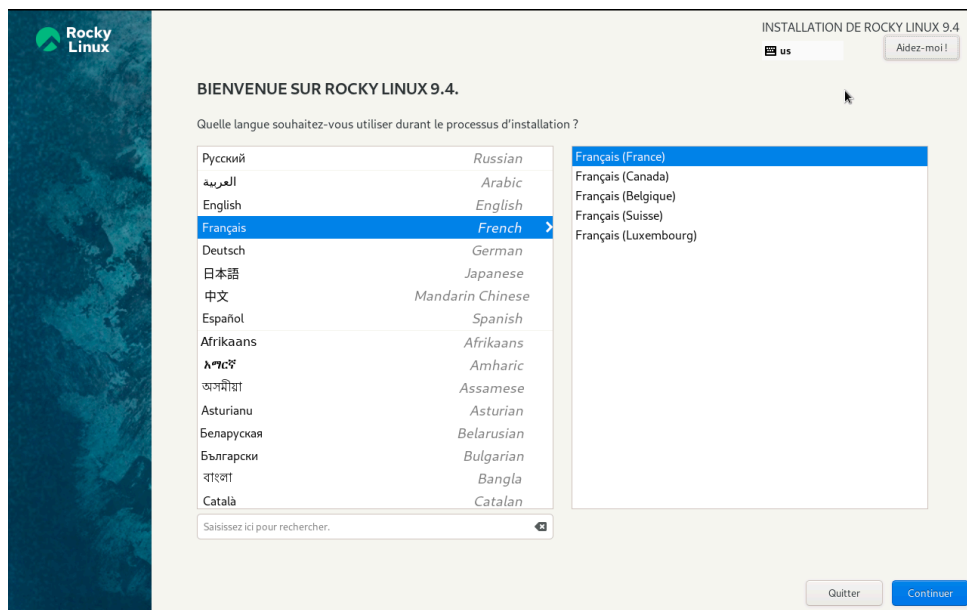
3) Виртуальной машине требуется оперативная память. Выделяем 4096 МБ, это половина основной оперативной памяти. Мы также даем ему 4 процессора.



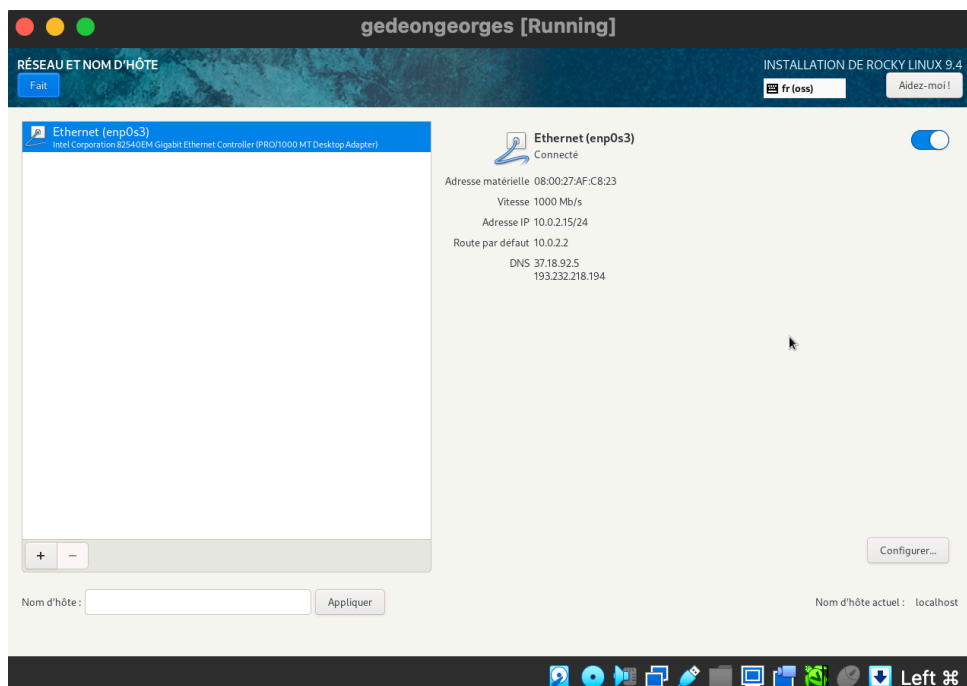
4)Выделяем под виртуальную машину 40 ГБ памяти.



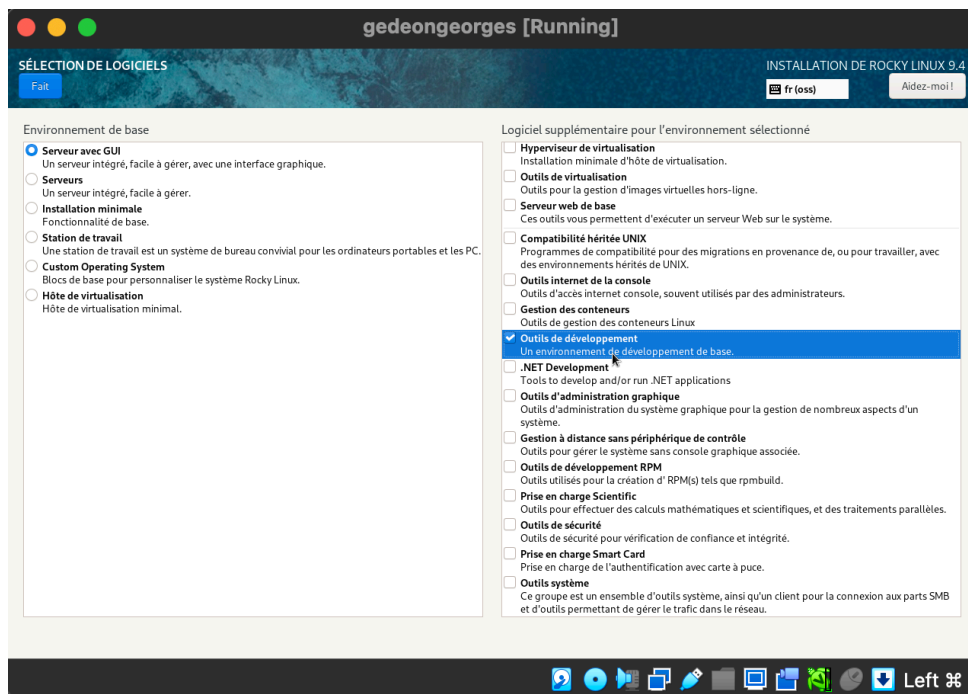
5)Включаем виртуальную машину, выбор языка.



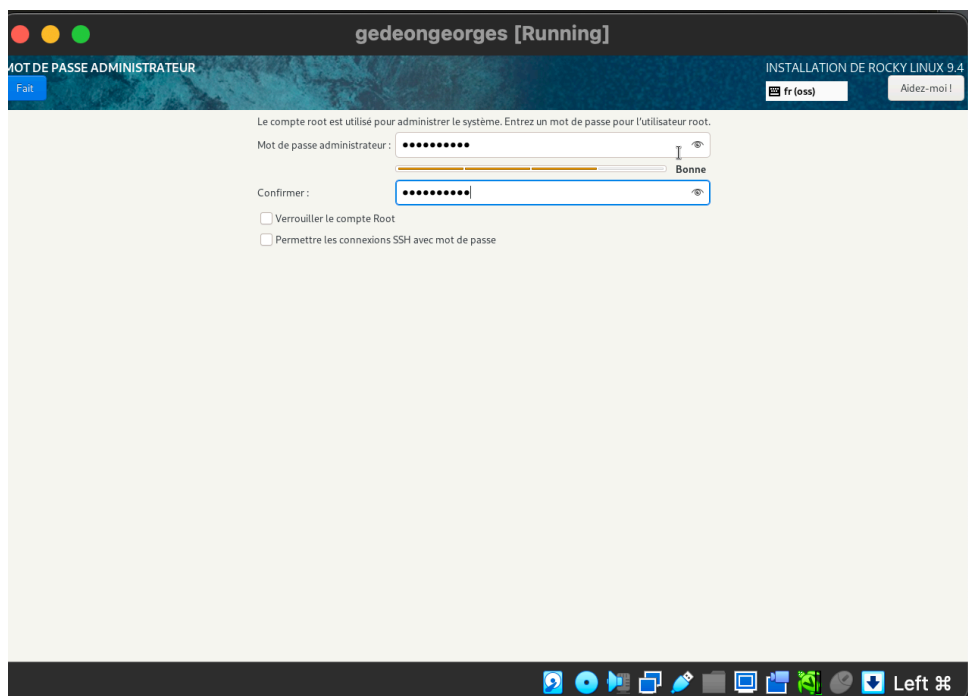
6) Загрузите сеть и следуйте за ней.



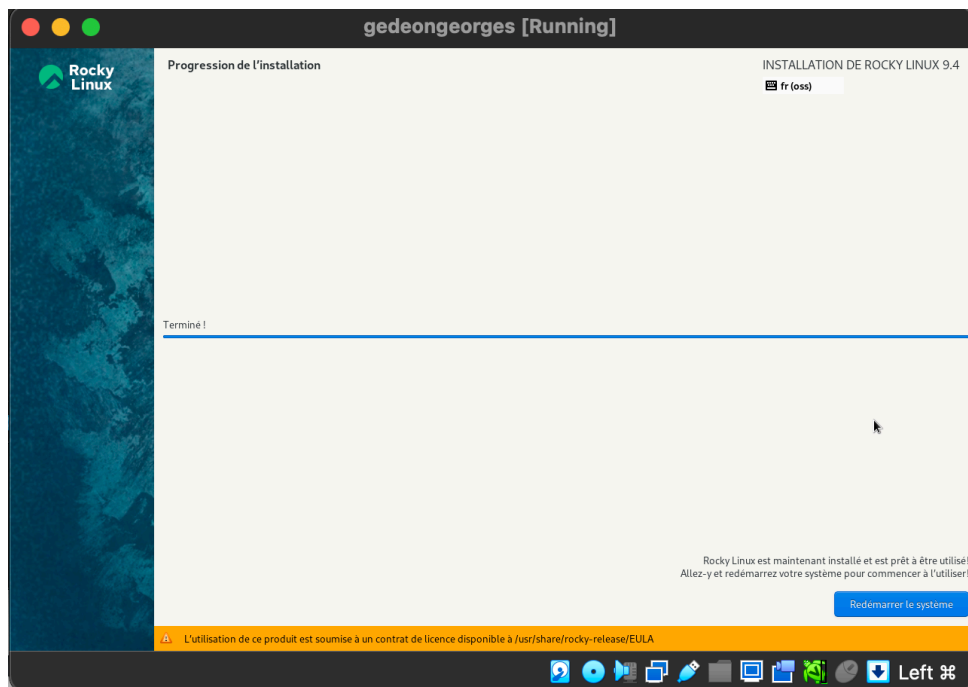
7) Базовое окружение выбираем Сервер с GUI.



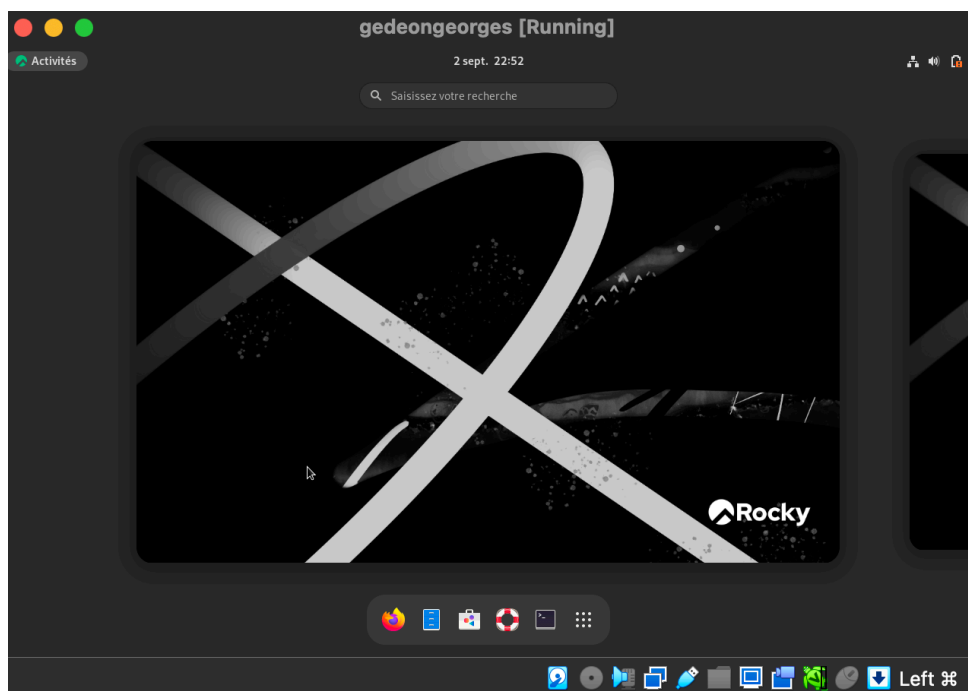
8) Установите пароль для пользователя root.



9) Установка завершена. Перезагрузить.



10)Создайте систему Rocky Linux. Установка завершена!



ДОМАШНЕЕ ЗАДАНИЕ №1:

1)Окно терминала подключается к системе, запустивкоманда dmesg.

```

0.000000 Command line: BOOT_IMAGE=(hd0,msdos1)/vmlinuz-5.14.0-427.13.1.el9_4.x86_64 root=/dev/mapper/rl-root ro crashkernel=1G-4G:192M,4G-64G:256M,
4G-512M resume=/dev/mapper/rl-swap rd.lvm.lv=rl/root rd.lvm.lv=rl/swap rhgb quiet
0.000000 x86/fpu: Supporting XSAVE feature 0x01: 'x87 floating point registers'
0.000000 x86/fpu: Supporting XSAVE feature 0x02: 'SSE registers'
0.000000 x86/fpu: Supporting XSAVE feature 0x04: '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 0x00000000-0x00000000-0x00000000-0x00000000] usable
0.000000 BIOS-e820: [mem 0x00000000-0x00000000-0x00000000-0x00000000] reserved
0.000000 BIOS-e820: [mem 0x00000000-0x00000000-0x00000000-0x00000000] reserved
0.000000 BIOS-e820: [mem 0x00000000-0x00000000-0x00000000-0x00000000] usable
0.000000 BIOS-e820: [mem 0x00000000-0x00000000-0x00000000-0x00000000] ACPI data
0.000000 BIOS-e820: [mem 0x00000000-0x00000000-0x00000000-0x00000000] reserved
0.000000 BIOS-e820: [mem 0x00000000-0x00000000-0x00000000-0x00000000] reserved
0.000000 BIOS-e820: [mem 0x00000000-0x00000000-0x00000000-0x00000000] reserved
0.000000 BIOS-e820: [mem 0x00000000-0x00000000-0x00000000-0x00000000] usable
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 kvm-clock: Using mrs 4b564d01 and 4b564d00
0.000000 kvm-clock: using sched offset of 2564155502676 cycles
0.000000 clocksource: kvm-clock: mask: 0xffffffffffffffff max_cycles: 0x1cd42e4dffb, max_idle_ns: 881590591483 ns
0.000000 tsc: Detected 2308.878 MHz processor
0.012085 e820: update [mem 0x00000000-0x00000000] usable ==> reserved
0.001294 e820: remove [mem 0x00000000-0x00000000] usable
0.001302 last_pfn = 0x120000 max_arch_pfn = 0x400000000
0.001370 MTRRs disabled by BIOS
0.001385 x86/PAT: Configuration [0-7]: WB WC UC- UC WB WP UC- WT
0.001527 last_pfn = 0x000000 max_arch_pfn = 0x400000000
0.001533 found SMP MP-table at [mem 0x00000000-0x00000000]
0.001749 RAMDISK: [mem 0x31048000-0x3481bfff]
0.001754 ACPI: Early table checksum verification disabled
0.001757 ACPI: RSDP 0x00000000-0x00000000 000024 (v02 VBOX )
0.001762 ACPI: XSDT 0x00000000-0x00000000 00003C (v01 VBOX VBOXXSDT 00000001 ASL 00000061)
0.001767 ACPI: FACP 0x00000000-0x00000000 0000F4 (v04 VBOX VBOXFACP 00000001 ASL 00000061)

```

Вы можете добавить ссылку на `grep:dmesg | grep -i "то, что мы ищем"` Команда `grep` используется для поиска данных.а. Версия ядра Linux (версия Linux).

```

[ggeorges@localhost ~]$ dmesg | grep -i "Linux version"
[ 0.000000] Linux version 5.14.0-427.13.1.el9_4.x86_64 (mockbuild@iadi-prod-build001.bld.equ.rockylinux.org) (gcc (GCC) 11.4.1 20231218 (Red Hat 11.4.1-3), GNU ld version 2.35.2-43.el9) #1 SMP PREEMPT_DYNAMIC Wed May 1 19:11:20 UTC 2024
[ggeorges@localhost ~]$

```

b. Частота процессора (Detected MHz processor).

```

[ggeorges@localhost ~]$ dmesg | grep -i "MHZ"
[ 0.000000] tsc: Detected 2308.878 MHz processor
[ 2.806284] e1000 0000:00:03:0 eth0: (PCI:33MHz:32-bit) 08:00:27:af:c8:23
[ggeorges@localhost ~]$

```

c. Модель процессора (CPU0)

```

[ggeorges@localhost ~]$ dmesg | grep -i "CPU0"
[ggeorges@localhost ~]$

```

d. Объем доступной оперативной памяти (Memory available)

```

[ggeorges@localhost ~]$ dmesg | grep -i "Memory"
[ 0.003788] ACPI: Reserving FACP table memory at [mem 0xdfff00f0-0xdfff01e3]
[ 0.003789] ACPI: Reserving DSDT table memory at [mem 0xdfff0620-0xdfff2972]
[ 0.001789] ACPI: Reserving FACS table memory at [mem 0xdfff0200-0xdfff023f]
[ 0.001790] ACPI: Reserving FACS table memory at [mem 0xdfff0200-0xdfff023f]
[ 0.001790] ACPI: Reserving APIC table memory at [mem 0xdfff0240-0xdfff02ab]
[ 0.001791] ACPI: Reserving SSDT table memory at [mem 0xdfff02b0-0xdfff061b]
[ 0.002906] Reserving 256MB of memory at 3312MB for crashkernel (System RAM: 4095MB)
[ 0.002924] Early memory node ranges
[ 0.010338] PM: hibernation: Registered nosave memory: [mem 0x00000000-0x00000fff]
[ 0.010340] PM: hibernation: Registered nosave memory: [mem 0x0000f000-0x0000ffff]
[ 0.010341] PM: hibernation: Registered nosave memory: [mem 0x000a0000-0x000effff]
[ 0.010341] PM: hibernation: Registered nosave memory: [mem 0x000f0000-0x000fffff]
[ 0.010342] PM: hibernation: Registered nosave memory: [mem 0xdfff0000-0xdfff0fff]
[ 0.010343] PM: hibernation: Registered nosave memory: [mem 0xe0000000-0xfebfffff]
[ 0.010343] PM: hibernation: Registered nosave memory: [mem 0xf0000000-0xf000ffff]
[ 0.010344] PM: hibernation: Registered nosave memory: [mem 0xf0000000-0xf000ffff]
[ 0.010344] PM: hibernation: Registered nosave memory: [mem 0xf0000000-0xf000ffff]
[ 0.010344] PM: hibernation: Registered nosave memory: [mem 0xf0000000-0xf000ffff]
[ 0.010345] PM: hibernation: Registered nosave memory: [mem 0xffff0000-0xffffffff]
[ 0.031800] Memory: 3414892K/4193848K available (16384K kernel code, 5626K rdata, 11748K rodata, 3892K init, 5956K bss, 510476K reserved, 0K cma-reserved)
[ 0.112767] Freeing SMP alternatives memory: 36K
[ 0.225938] x86/mm: Memory block size: 128MB
[ 0.446718] Non-volatile memory driver v1.3
[ 0.947273] Freeing initrd memory: 57168K
[ 1.272408] Freeing unused decrypted memory: 2028K
[ 1.272409] Freeing unused kernel image (initmem) memory: 3892K
[ 1.279104] Freeing unused kernel image (rodata/data gap) memory: 540K
[ 2.359645] vmwgfx 0000:00:02:0: [drm] Legacy memory limits: VRAM = 16384 kB, FIFO = 2048 kB, surface = 507904 kB
[ 2.359650] vmwgfx 0000:00:02:0: [drm] Maximum display memory size is 16384 KiB
[ggeorges@localhost ~]$

```

e. Тип обнаруженного гипервизора (Hypervisor detected).

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

f. Тип файловой системы корневого раздела. Последовательность монтирования файловых систем

```
[ggeorges@localhost ~]$ dmesg | grep -i "Mount"
[ 0.099778] Mount-cache hash table entries: 8192 (order: 4, 65536 bytes, linear)
[ 0.099778] Mountpoint-cache hash table entries: 8192 (order: 4, 65536 bytes, linear)
[ 4.285938] XFS (dm-0): Mounting V5 Filesystem 2020d098-018e-4bff-a410-aa13d910fbc8
[ 4.318355] XFS (dm-0): Ending clean mount
[ 5.487723] systemd[1]: Set up automount Arbitrary Executable File Formats File System Automount Point.
[ 5.513005] systemd[1]: Mounting Huge Pages File System...
[ 5.514887] systemd[1]: Mounting POSIX Message Queue File System...
[ 5.517547] systemd[1]: Mounting Kernel Debug File System...
[ 5.520576] systemd[1]: Mounting Kernel Trace File System...
[ 5.580335] systemd[1]: Starting Remount Root and Kernel File Systems...
[ 5.593904] systemd[1]: Mounted Huge Pages File System.
[ 5.594364] systemd[1]: Mounted POSIX Message Queue File System.
[ 5.594619] systemd[1]: Mounted Kernel Debug File System.
[ 5.595146] systemd[1]: Mounted Kernel Trace File System.
[ 5.619679] systemd[1]: Mounting FUSE Control File System...
[ 5.624811] systemd[1]: Mounting Kernel Configuration File System...
[ 7.320167] XFS (sda1): Mounting V5 Filesystem b173229b-baba-4e2e-bee2-2bb8cd7ea27c
[ 7.799743] XFS (sda1): Ending clean mount
[ggeorges@localhost ~]$
```

КОНТРОЛЬНЫЕ ВОПРОСЫ №1

1) Учётная запись пользователя содержит сведения, необходимые для идентификации пользователя при подключении к системе, такие как имя пользователя, имя хоста и пароль.

2) Команды терминала: а. Для получения справки используется ключ – help или команда man. Например, ls –help или man ls. б. Для перемещения по файловой системе используется команда cd. Например cd ~. в. Для просмотра содержимого каталога используется команда ls. Например ls ~/work. г. Для определения объёма каталога используется команда du. е. Для создания каталогов используется mkdir, для удаления пустых каталогов используется rmdir. Для создания файлов используется touch, для удаления файлов и каталогов используется rm. ф. Для задания прав используется команда chmod. Например, chmod u-w test.txt. г. Для просмотра истории команд используется команда history.

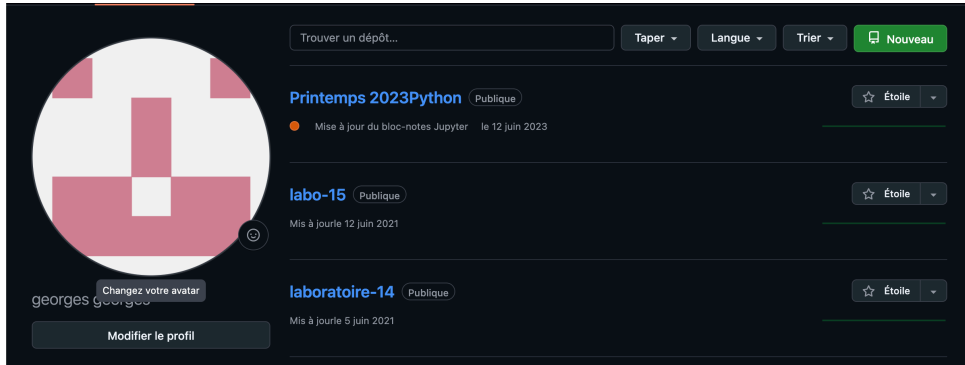
3) Файловая система — часть ОС, которая обеспечивает чтение и запись файлов на дисковых носителях информации. а. Ext2 — расширенная файловая система. Данные сначала кэшируются и только потом записываются на диск. б. Ext3 и Ext4 — журналируемые файловые системы. Осуществляется хранение в виде журнала со списком изменений, что помогает сохранить целостность при сбоях. в. XFS — высокопроизводительная журналируемая файловая система, рассчитанная для работы на дисках большого объёма.

4) Для просмотра подмонтированных в ОС файловых систем необходимо использовать команду findmnt.

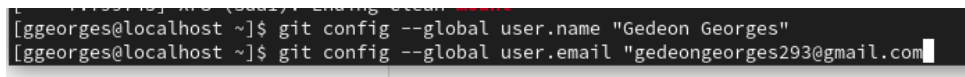
5) Для удаления зависшего процесса используется команда `kill PID` или `killall название`.

Часть 2. Работа с GitHub

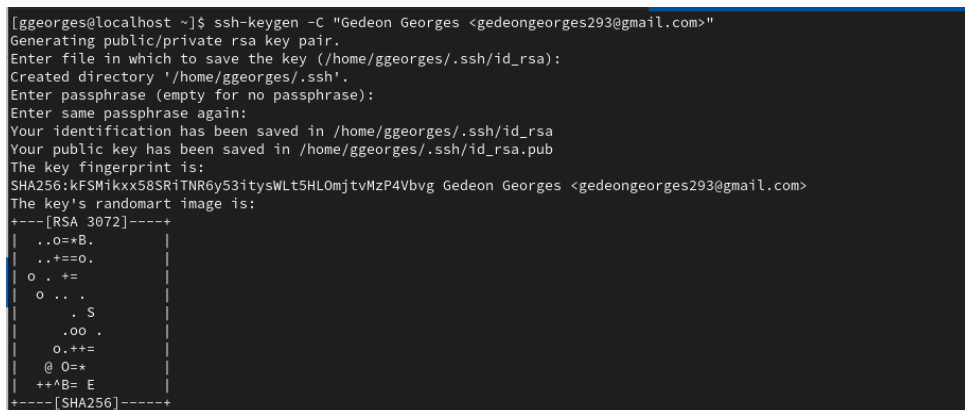
1) Первым делом, мы регистрируемся на сайте github.com. Так как мы не в первый раз имеем дело с github, то аккаунт уже готов.



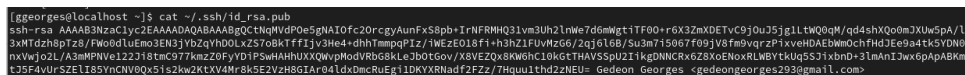
2) Первоначальная настройка для работы с github.



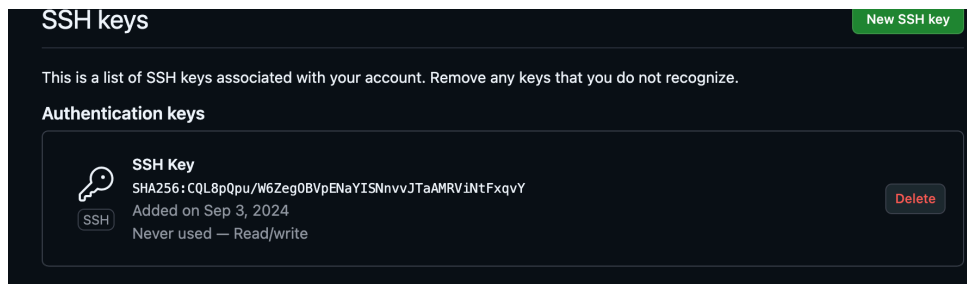
3) Генерация SSH ключа.



4) Переходим на сайте в настройки и в SSH Keys. Копируем наш SSH ключ, предварительно выведя его в консоль с помощью `cat ~/.ssh/id_rsa.pub`.



5) Ключ отобразился на сайте.



КОНТРОЛЬНЫЕ ВОПРОСЫ №2:

1) Система контроля версий Git представляет собой набор программ командной строки. Доступ к ним можно получить из терминала посредством ввода команды `git` различными опциями. Системы контроля версий (Version Control System, VCS) применяются при работе нескольких человек над одним проектом.

2) В классических системах контроля версий используется централизованная модель, предполагающая наличие единого репозитория для хранения файлов. Выполнение большинства функций по управлению версиями осуществляется специальным сервером. Участник проекта (пользователь) перед началом работы посредством определённых команд получает нужную ему версию файлов. После внесения изменений, пользователь размещает новую версию в хранилище. При этом предыдущие версии не удаляются из центрального хранилища и к ним можно вернуться в любой момент. Сервер может сохранять неполную версию изменённых файлов, а производить так называемую дельта-компрессию — сохранять только изменения между последовательными версиями, что позволяет уменьшить объём хранимых данных. Системы контроля версий также могут обеспечивать дополнительные, более гибкие функциональные возможности. Например, они могут поддерживать работу с несколькими версиями одного файла, сохраняя общую историю изменений до точки ветвления версий и собственные истории изменений каждой ветви. Кроме того, обычно доступна информация о том, кто из участников, когда и какие изменения вносил. Обычно такого рода информация хранится в журнале изменений, доступ к которому можно ограничить.

3) Централизованные системы — это системы, которые используют архитектуру клиент / сервер, где один или несколько клиентских узлов напрямую подключены к центральному серверу. Пример — Wikipedia. В децентрализованных системах каждый узел принимает свое собственное решение. Конечное поведение системы является

совокупностью решений отдельных узлов. Пример — Bitcoin. В классических системах контроля версий используется централизованная модель, предполагающая наличие единого репозитория для хранения файлов. Выполнение большинства функций по управлению версиями осуществляется специальным сервером.

4) Создадим локальный репозиторий. Сначала сделаем предварительную конфигурацию, указав имя и email владельца репозитория: `git config --global user.name "Имя Фамилия"` `git config --global user.email "Project24@mail"` и настроив utf-8 в выводе сообщений: `git config --global quotepath false` Для инициализации локального репозитория, расположенного, например, в каталоге `~/tutorial`, необходимо ввести в командной строке: `cd` `mkdir tutorial` `cd tutorial` `git init`

5) Для последующей идентификации пользователя на сервере репозитория необходимо сгенерировать пару ключей (приватный и открытый): `ssh-keygen -C "Имя Фамилия Project24@mail"` Ключи сохраняться в каталоге `~/.ssh/`. Скопировав из локальной консоли ключ в буфер обмена: `cat ~/.ssh/id_rsa.pub | xclip -sel clip` вставляем ключ в появившееся на сайте поле.

6) У Git две основных задачи: первая — хранить информацию о всех изменениях в вашем коде, начиная с самой первой строчки, а вторая — обеспечение удобства командной работы над кодом.

7) Основные команды git: Наиболее часто используемые команды git: — создание основного дерева репозитория: `git init` — получение обновлений (изменений) текущего дерева из центрального репозитория: `git pull` — отправка всех произведённых изменений локального дерева в центральный репозиторий: `git push` — просмотр списка изменённых файлов в текущей директории: `git status` — просмотр текущих изменений: `git diff` — сохранение текущих изменений: — добавить все изменённые и/или созданные файлы и/или каталоги: `git add .` — добавить конкретные изменённые и/или созданные файлы и/или каталоги: `git add имена_файлов` — удалить файл и/или каталог из индекса репозитория (при этом файл и/или каталог остаётся в локальной директории): `git rm имена_файлов` — сохранение добавленных изменений: — сохранить все добавленные изменения и все изменённые файлы: `git commit -am 'Описание коммита'` — сохранить добавленные изменения с внесением комментария через встроенный редактор: `git commit` — создание новой ветки, базирующейся на текущей: `git checkout -b имя_ветки` — переключение

на некоторую ветку: `git checkout имя_ветки` (при переключении на ветку, которой ещё нет в локальном репозитории, она будет создана и связана с удалённой) – отправка изменений конкретной ветки в центральный репозиторий: `git push origin имя_ветки`–слияние ветки стекущим деревом: `git merge --no-ff имя_ветки`–удаление ветки: – удаление локальной уже слитой с основным деревом ветки: `git branch -d имя_ветки`–принудительное удаление локальной ветки: `git branch -D имя_ветки`–удаление ветки с центрального репозитория: `git push origin :имя_ветки`

8)Использования git при работе с локальными репозиториями (добавления текстового документа в локальный репозиторий):`git add hello.txt``git commit -am'Новый файл'`

9)Проблемы, которые решают ветки git:• нужно постоянно создавать архивы с рабочим кодом• сложно "переключаться" между архивами• сложно перетаскивать изменения между архивами• легко что-то напутать или потерять

10)Во время работы над проектом так или иначе могут создаваться файлы, которые не требуется добавлять в последствии в репозиторий. Например, временные файлы, создаваемые редакторами, или объектные файлы, создаваемые компиляторами. Можно прописать шаблоны игнорируемых при добавлении в репозиторий типов файлов в файл .gitignore с помощью ервисов. Для этого сначала нужно получить списки имеющихся шаблонов: `curl -L -s https://www.gitignore.io/api/list`Затем скачать шаблон, например, для C и C++`curl -L -s https://www.gitignore.io/api/c >> .gitignore``curl -L -s https://www.gitignore.io/api/c++ >> .gitignore`

Выводы

Я установил VirtualBox, поставил на виртуальную машину Rocky Linux. Скомпилировал файлы из формата md в pdf и docx.