Отчёт по лабораторной работе №2

Дисциплина: Архитектура Компьютеров и Операционные Системы

Дауд Амжад

Содержание

Список иллюстраций

1 Цель работы

Изучение идеалогии, применение средств контроля версий и освоение умения по работе с git.

2 Задание

1. Создать базовую конфигурацию для работы с git.
2. Создать ключ SSH.
3. Создать ключ PGP.
4. Настроить подписи git.
5. Зарегистрироваться на Github.
6. Создать локальный каталог для выполнения заданий по предмету.

3 Выполнение лабораторной работы

3.1 Создание базовой конфигурации для работы с git.

Установливаю git используя “dnf install git”:

```
[amjaddawud@vbox ~]$ sudo -i
[root@vbox ~]# dnf install git
Updating and loading repositories:
Repositories loaded.
Package "git-2.48.1-1.fc41.x86_64" is already installed.
```

Рис. 3.1: Установление git

С помощью dnf install gh, установливаю gh:

```
[root@vbox ~]# dnf install gh
Updating and loading repositories:
Repositories loaded.
Package             Arch   Version             Repository       Size
Installing:
 gh                 x86_64 2.65.0-1.fc41       updates      42.6 MiB

Transaction Summary:
 Installing:         1 package

Total size of inbound packages is 10 MiB. Need to download 10 MiB.
After this operation, 43 MiB extra will be used (install 43 MiB, remove 0 B).
Is this ok [y/N]: y
```

Рис. 3.2: Установление gh

В качестве имя и email владельца репозитории задаю свои имя и email и настраиваю utf-8:

```
[amjaddawud@vbox ~]$ git config --global user.name "amjaddawud"
[amjaddawud@vbox ~]$ git config --global user.email "1032245416@pfur.ru"
[amjaddawud@vbox ~]$ git config --global core.quotepath false
```

Рис. 3.3: имя и email владельца

Задаю имя начальной ветки и паррамеры autocrlf и safecrlf:

```
[amjaddawud@vbox ~]$ git config --global init.defaultBranch master
[amjaddawud@vbox ~]$ git config --global core.autocrlf input
[amjaddawud@vbox ~]$ git config --global core.safecrlf warn
```

Рис. 3.4: имя начальной ветки и паррамеры

3.2 Создание ключ ssh

Создаю ключи ssh по алгоритму rsa с размером 4096 бит:

```
[amjaddawud@vbox ~]$ ssh-keygen -t rsa -b 4096
Generating public/private rsa key pair.
Enter file in which to save the key (/home/amjaddawud/.ssh/id_rsa):
/home/amjaddawud/.ssh/id_rsa already exists.
Overwrite (y/n)? y
Enter passphrase for "/home/amjaddawud/.ssh/id_rsa" (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/amjaddawud/.ssh/id_rsa
Your public key has been saved in /home/amjaddawud/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:9wmUCJUWk19hnOvuWI3jz2oceb57L5uQF1oAqTfC8bo amjaddawud@vbox
The key's randomart image is:
+---[RSA 4096]----+
|      ooo+++     |
|      .o+o=o     |
|       +.=o..    |
|        =.=  .   |
|        S=oo  o  |
|        ..+=.= . |
|         +++B .  |
|        E++o.oo. |
|        .o+o=+o+.|
+----[SHA256]-----+
```

Рис. 3.5: Создание ключ ssh

3.3 Создание ключ gpg

Генерирую ключ gpg –full-generate-key:

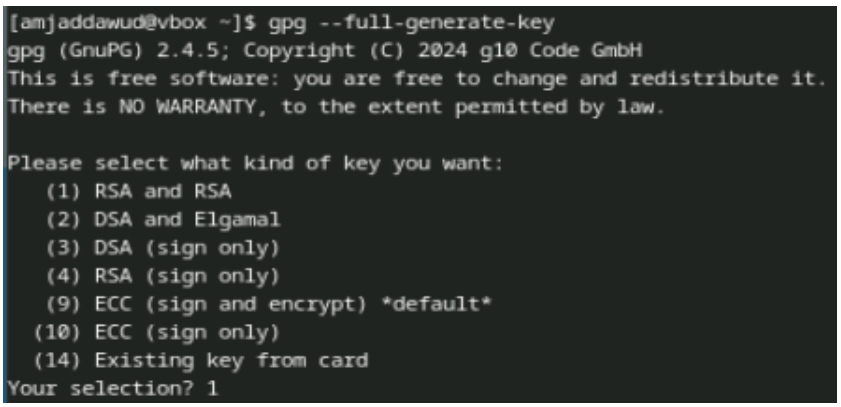

Рис. 3.6: Создание ключ gpg

Из предложенных опций выбираю тип RSA and RSA; размер 4096; срок действия 0:

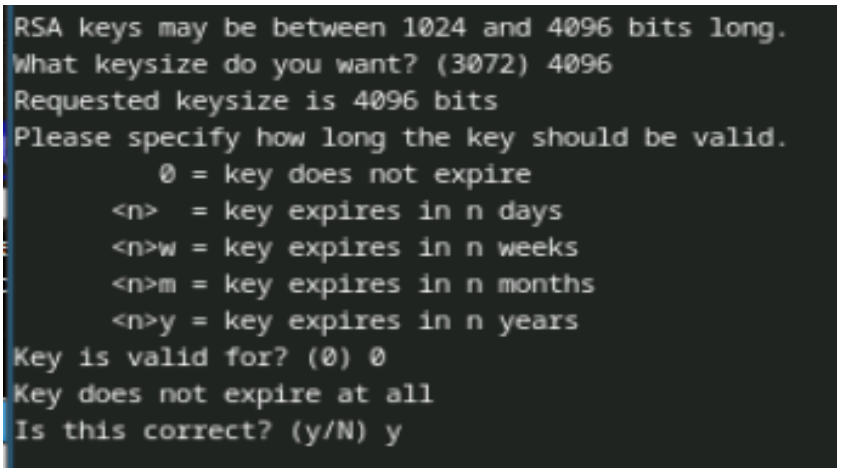

Рис. 3.7: Настройки ключ gpg

GPG запросил личную информацию, которая сохранится в ключе Имя и адрес электронной почты:

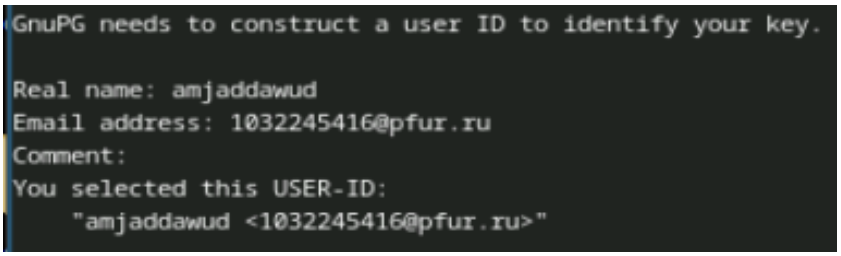

Рис. 3.8: личная информация

У меня уже есть аккаунт на github, поэтому я вхожу в систему:

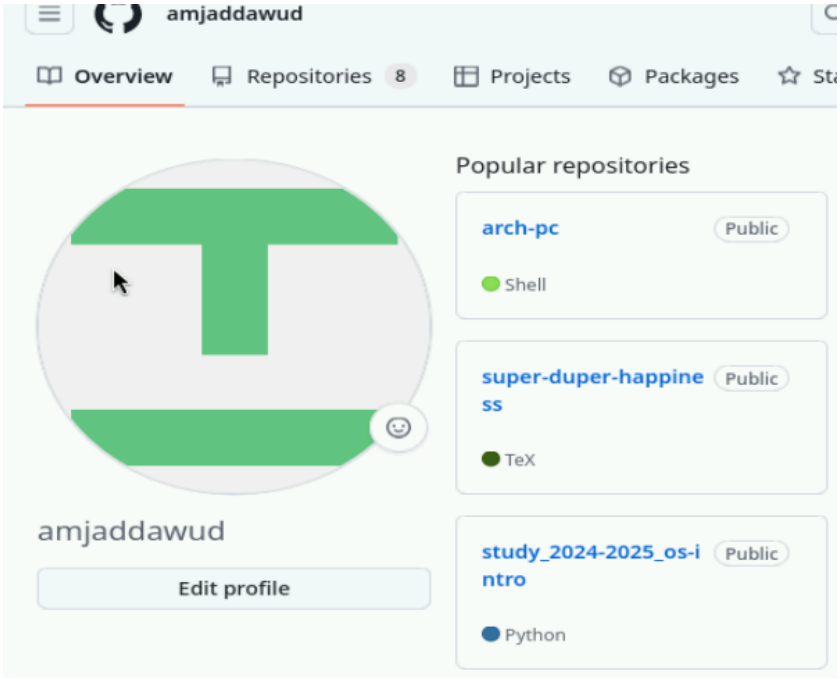

Рис. 3.9: аккаунт на git

Вывожу список ключей:

```
[amjaddawud@vbox ~]$ gpg --list-secret-keys --keyid-format LONG
gpg: checking the trustdb
gpg: marginals needed: 3  completes needed: 1  trust model: pgp
gpg: depth: 0  valid:   2  signed:   0  trust: 0-, 0q, 0n, 0m, 0f, 2u
[keyboxd]
---------
sec   rsa4096/15450EC691F6DDCD 2025-03-08 [SC]
      4BD4677D9CB7F54409D002C115450EC691F6DDCD
uid                 [ultimate] amjaddawud <1032245416@pfur.ru>
ssb   rsa4096/14DD87A42A58D41C 2025-03-08 [E]

sec   rsa4096/D5A8AB1D389333F3 2025-03-08 [SC]
      EB17C60EFD5D64E5E776D440D5A8AB1D389333F3
uid                 [ultimate] amjaddawud <1032245416@pfur.ru>
ssb   rsa4096/8A3E2B0723324EC0 2025-03-08 [E]
```

Рис. 3.10: список ключей

Установливаю xclip:

```
[amjaddawud@vbox ~]$ sudo dnf install xclip
[sudo] password for amjaddawud:
Updating and loading repositories:
Repositories loaded.
Package             Arch   Version              Repository      Size
Installing:
 xclip              x86_64 0.13-22.git11cba61.f fedora      62.4 KiB

Transaction Summary:
 Installing:         1 package
```

Рис. 3.11: Установление xclip

Скопирую сгенерированный gpg ключ в буфер обмена:

```
[amjaddawud@vbox ~]$ git config --global user.signingkey D5A8AB1D389333F3
[amjaddawud@vbox ~]$ git config --global commit.gpgsign true
[amjaddawud@vbox ~]$ git config --global gpg.program $(which gpg2)
[amjaddawud@vbox ~]$
```

Рис. 3.12: Копирование ключ gpg

Далее перехожу в настройки GitHub, нажимаю на кнопку New GPG key и вставляю полученный ключ:

```
[amjaddawud@vbox ~]$ gh auth login
? Where do you use GitHub? GitHub.com
? What is your preferred protocol for Git operations on this host? SSH
? Upload your SSH public key to your GitHub account? /home/amjaddawud/.ssh/id_ed25519.pub
? Title for your SSH key: GitHub CLI
? How would you like to authenticate GitHub CLI? Login with a web browser

! First copy your one-time code: 054E-6739
```

Рис. 3.13: Добавлен ключ gpg

Используя введёный email, указываю Git применять его при подписи коммитов:

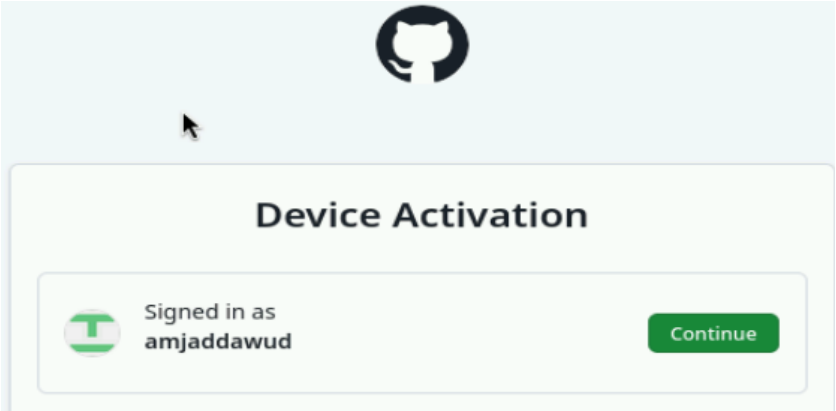

Рис. 3.14: указиваю Git

Начинаю авторизацию в gh используя gh auth login:

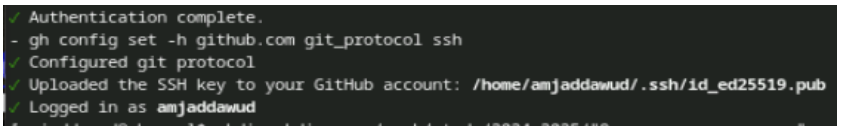

Рис. 3.15: авторизацию в gh

Завершаю авторизацию на броузер:

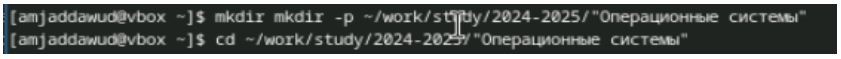

Рис. 3.16: Авторизоваться через броузер.

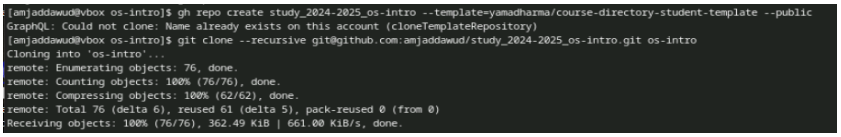

Рис. 3.17: Завершена авторизация

3.4 Создание локального каталога для выполнения заданий.

Создаю каталог “mkdir -p ~/work/study/2022-2023/”Операционные системы”:

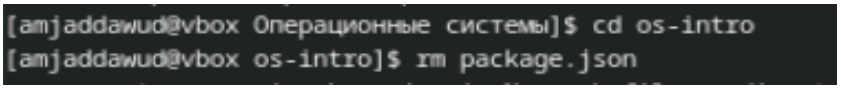

Рис. 3.18: Создание каталог

Перехожу в созданный каталог:

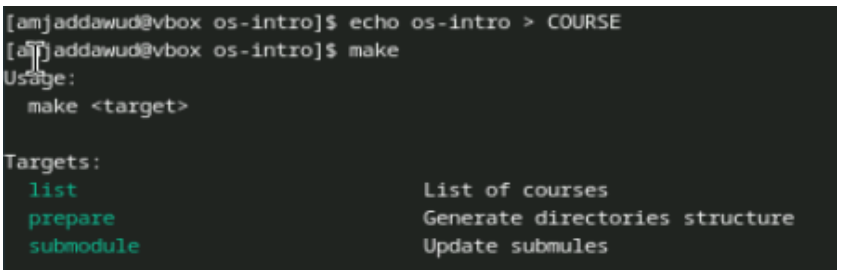

Рис. 3.19: Создание каталог

Удаляю лишные файлы:

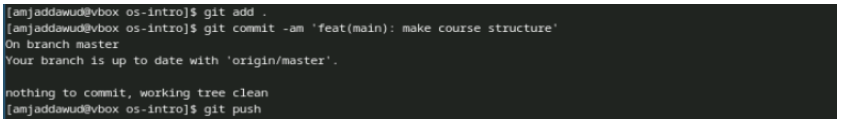

Рис. 3.20: Удаление файла

Создаю еще необходимые каталоги:

Создани необходимых каталогов

Рис. 3.21: Создани необходимых каталогов

Отправляю Файлы на сервер:

Отправление файлы на сервер

Рис. 3.22: Отправление файлы на сервер

4 Выводы

При выполнении лабораторной работы я изучила идеалогию, применение средств контроля версий и освоеила умение по работе с git.

5 Ответы на контрольные вопросы

1. Системы Контроля Версий - Программные инструменты, помагающие командам разработчиков управлять изменениями в исходном коде с течением времени.

2. Хранилище - в нем храняются все документы, включая историю их изменение и прочей служебной информацией.

   commit - отслеживание измененийб сохраняет разницу в изменениях.

   история - Хранит все изменения в проекте и позволяет при необходимости обратиться к нужным данным.

   рабочая копия- копия проекта основанная на версии из хранилища.

3. В ценрализованном VCS например AccuRev, каждый пользователь копирует себе необходимые ему файлы из репзитория, изменяет их а затем добавляет измениения обратно в хранилище. В децентрализованном VCS например Git, есть возможность добавлять и забирать изменения из любого репозитория.

4. Сначала создается и подключается удаленный репозиторий, затем по мере изменения проекта эти изменения отправляются на сервер.

5. Участник проекта перед началом работы получает нужную ему версию проекта в хранилище, с помощью определенных команд, после внесения изменений пользователь размещает новую версию в хранилище. При этом предыдущие версии не удаляются. К ним можно вернуться в любой момент.

6. Хранение информации о всех изменениях в вашем коде, обеспечение удобства командной работы над кодом.

Создание основного дерева репозитория: git init

7. 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 с текущим 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. git push -all отправляем из локального репозитория все сохраненные изменения в центральный репозиторий, предварительно создав локальный репозиторий и сделав предварительную конфигурацию.

9. Ветвление - один из параллельных участков в одном хранилище, исходящих из одной версии, обычно есть главная ветка. Между ветками, т. е. их концами возможно их слияние. Используются для разработки новых функций.

10. Во время работы над проектом могут создаваться файлы, которые не следуют добавлять в репозиторий. Например, временные файлы. Можно прописать шаблоны игнорируемых при добавлении в репозиторий типов файлов в файл .gitignore с помощью сервисов.

6 Список литературы

::: Архитектура ЭВМ :::