

Отчет по лабораторной работе №2

Операционные системы

Зарицкая Марина Петровна

Содержание

1	Цель работы	5
2	Задание	6
3	Выполнение лабораторной работы	7
3.1	Установка программного обеспечения	7
3.2	Базовая настройка git	7
3.3	Создание ключа SSH	8
3.4	Создание ключа GPG	9
3.5	Регистрация на Github	11
3.6	Добавление ключа GPG в Github	11
3.7	Настроить подписи Git	13
3.8	Настройка gh	14
3.9	Создание репозитория курса на основе шаблона	15
4	Выводы	16
5	Ответы на контрольные вопросы.	17
	Список литературы	20

Список иллюстраций

3.1	Установка git и gh	7
3.2	Задаю имя и email владельца репозитория	8
3.3	Настройка utf-8 в выводе сообщений git	8
3.4	Задаю имя начальной ветки	8
3.5	Задаю параметры autocrlf и safecrlf	8
3.6	Генерация ssh ключа по алгоритму rsa	9
3.7	Генерация ssh ключа по алгоритму ed25519	9
3.8	Генерация ключа	10
3.9	Защита ключа GPG	10
3.10	Аккаунт на Github	11
3.11	Вывод списка ключей	12
3.12	Копирование ключа в буфер обмена	12
3.13	Настройки GitHub	12
3.14	Добавление нового PGP ключа	13
3.15	Добавленный ключ GPG	13
3.16	Настройка подписей Git	14
3.17	Авторизация в gh	14
3.18	Завершение авторизации через браузер	14
3.19	Завершение авторизации	15

Список таблиц

1 Цель работы

Цель данной лабораторной работы – изучение идеологии и применения средств контроля версий, освоение умения по работе с git.

2 Задание

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

3 Выполнение лабораторной работы

3.1 Установка программного обеспечения

Устанавливаю необходимое программное обеспечение git и gh через терминал с помощью команд: `dnf install git` и `dnf install gh` (рис. 3.1).

```
mpzarickaya@mpzarickaya:~$ sudo dnf install git
[sudo] password for mpzarickaya:
Copr repo for gitflow owned by elegos
Copr repo for browserpass owned by maximbaz
Copr repo for iosevka owned by peterwu
Copr repo for iosevka owned by peterwu
Fedora 39 - x86_64
Fedora 39 openh264 (From Cisco) - x86_64
Fedora 39 - x86_64 - Updates
Fedora 39 - x86_64 - Updates
Package git-2.44.0-1.fc39.x86_64 is already installed.
Dependencies resolved.
Nothing to do.
Complete!
mpzarickaya@mpzarickaya:~$ sudo dnf install gh
Last metadata expiration check: 0:03:51 ago on Fri 16 Aug 2024 21:39:05 MSK.
Package gh-2.45.0-1.fc39.x86_64 is already installed.
Dependencies resolved.
Nothing to do.
Complete!
mpzarickaya@mpzarickaya:~$
```

Рис. 3.1: Установка git и gh

3.2 Базовая настройка git

Задаю в качестве имени и email владельца репозитория свои имя, фамилию и электронную почту (рис. 3.2).

```
mpzarickaya@mpzarickaya:~$ git config --global user.name "Marina Zaritskaya"
mpzarickaya@mpzarickaya:~$ git config --global user.email "1132236026@pfur.ru"
mpzarickaya@mpzarickaya:~$
```

Рис. 3.2: Задаю имя и email владельца репозитория

Настраиваю utf-8 в выводе сообщений git для их корректного отображения (рис. 3.3).

```
mpzarickaya@mpzarickaya:~$ git config --global core.quotepath false
mpzarickaya@mpzarickaya:~$
```

Рис. 3.3: Настройка utf-8 в выводе сообщений git

Начальной ветке задаю имя master (рис. 3.4).

```
mpzarickaya@mpzarickaya:~$ git config --global init.defaultBranch master
mpzarickaya@mpzarickaya:~$
```

Рис. 3.4: Задаю имя начальной ветки

Задаю параметры autocrlf и safecrlf для корректного отображения конца строки (рис. 3.5).

```
mpzarickaya@mpzarickaya:~$ git config --global core.autocrlf input
mpzarickaya@mpzarickaya:~$ git config --global core.safecrlf warn
mpzarickaya@mpzarickaya:~$
```

Рис. 3.5: Задаю параметры autocrlf и safecrlf

3.3 Создание ключа SSH

Создаю ключ ssh размером 4096 бит по алгоритму rsa (рис. 3.6).


```

mpzarickaya@mpzarickaya:~$ ssh-keygen -t rsa -b 4096
Generating public/private rsa key pair.
Enter file in which to save the key (/home/mpzarickaya/.ssh/id_rsa):
/home/mpzarickaya/.ssh/id_rsa already exists.
Overwrite (y/n)? y
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/mpzarickaya/.ssh/id_rsa
Your public key has been saved in /home/mpzarickaya/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:IdIDAuuQk2JrGJXMkqMoroN62yX7xPPs0eQ3x5TMEYc mpzarickaya@mpzarickaya.mpzarickaya.net
The key's randomart image is:
+---[RSA 4096]-----+
|.o..      . |
|+=+ o      E . |
|@+ . + .   o |
|Xo. . o .   |
|++      S o o |
|.. . . =     |
|o . =o o     |
|+ .. = +o.o o |
|oo..o..=o o  |
+---[SHA256]-----+
mpzarickaya@mpzarickaya:~$

```

Рис. 3.6: Генерация ssh ключа по алгоритму rsa

Создаю ключ ssh по алгоритму ed25519 (рис. 3.7).

```

mpzarickaya@mpzarickaya:~$ ssh-keygen -t ed25519
Generating public/private ed25519 key pair.
Enter file in which to save the key (/home/mpzarickaya/.ssh/id_ed25519):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/mpzarickaya/.ssh/id_ed25519
Your public key has been saved in /home/mpzarickaya/.ssh/id_ed25519.pub
The key fingerprint is:
SHA256:WTZbf1KiYQ1WijkuEwPBCERh6C82apwI/VC7uEg9A mpzarickaya@mpzarickaya.mpzarickaya.net
The key's randomart image is:
+--[ED25519 256]--+
|*+=+o.      o.. |
|+o+.      + + |
|.o .o      B = o . |
| o.o + . = * + o |
|+.E o o S o . o . |
|=+ .      . o |
|* + .      |
| * o .      |
|. +..      |
+---[SHA256]-----+
mpzarickaya@mpzarickaya:~$

```

Рис. 3.7: Генерация ssh ключа по алгоритму ed25519

3.4 Создание ключа GPG

Генерирую ключ GPG, затем выбираю тип ключа RSA and RSA, задаю максимальную длину ключа: 4096, оставляю неограниченный срок действия ключа.

Далее отвечаю на вопросы программы о личной информации (рис. 3.8).

```
mpzarickaya@mpzarickaya:~$ gpg --full-generate-key
gpg (GnuPG) 2.4.4; Copyright (C) 2024 g10 Code GmbH
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.

Please select what kind of key you want:
  (1) RSA and RSA
  (2) DSA and Elgamal
  (3) DSA (sign only)
  (4) RSA (sign only)
  (9) ECC (sign and encrypt) *default*
  (10) ECC (sign only)
  (14) Existing key from card
Your selection? 1
RSA keys may be between 1024 and 4096 bits long.
What keysize do you want? (3072) 4096
Requested keysize is 4096 bits
Please specify how long the key should be valid.
  0 = key does not expire
  <n> = key expires in n days
  <n>w = key expires in n weeks
  <n>m = key expires in n months
  <n>y = key expires in n years
Key is valid for? (0) 0
Key does not expire at all
Is this correct? (y/N) y

GnuPG needs to construct a user ID to identify your key.

Real name: ZaritskayaMarina
Email address: 1132236026@pfur.ru
Comment:
You selected this USER-ID:
  "ZaritskayaMarina <1132236026@pfur.ru>"
```

Рис. 3.8: Генерация ключа

Ввожу фразу-пароль для защиты нового ключа (рис. 3.9).

A screenshot of a terminal window showing a GPG passphrase prompt. The prompt text is "Please enter the passphrase to protect your new key". Below it, the label "Passphrase:" is followed by a series of asterisks "*****" and a cursor. At the bottom of the prompt area, there are two buttons: "<OK>" on the left and "<Cancel>" on the right. The entire prompt is enclosed in a rectangular box.

Рис. 3.9: Защита ключа GPG

3.5 Регистрация на Github

У меня уже был создан аккаунт на Github, соответственно, основные данные аккаунта я так же заполняла и проводила его настройку, поэтому просто вхожу в свой аккаунт (рис. 3.10).

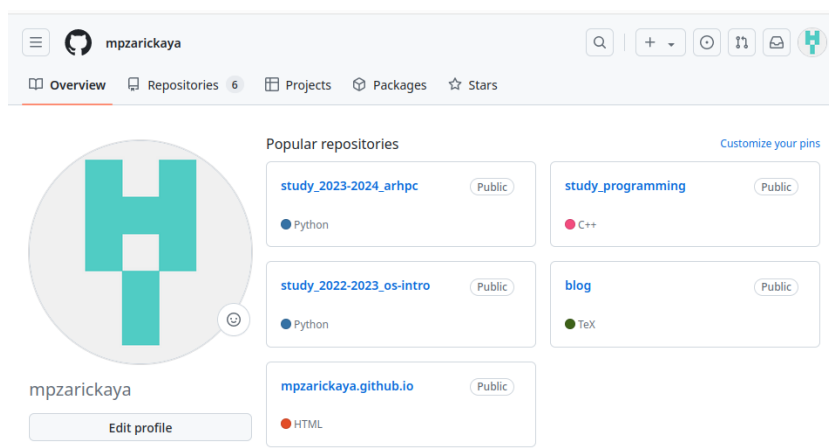


Рис. 3.10: Аккаунт на Github

3.6 Добавление ключа GPG в Github

Вывожу список созданных ключей в терминал, ищу в результате запроса отпечаток ключа (последовательность байтов для идентификации более длинного, по сравнению с самим отпечатком, ключа), он стоит после знака слеша, копирую его в буфер обмена (рис. 3.11).

```
mpzarickaya@mpzarickaya:~$ 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: 3 signed: 0 trust: 0-, 0q, 0n, 0m, 0f, 3u
[keyboard]
-----
sec   rsa4096/3D062ADB565EDC58 2024-03-16 [SC]
      6F8D3C161683604BE5D04E283D062ADB565EDC58
uid   [ultimate] Марина <1132236026@pfur.ru>
ssb   rsa4096/D1F94F9821CB3768 2024-03-16 [E]

sec   rsa4096/FE33C74D978EE751 2024-08-16 [SC]
      B1521B4A77C0C2EC8E07A474FE33C74D978EE751
uid   [ultimate] ZaritskayaMarina <1132236026@pfur.ru>
ssb   rsa4096/E3BCCBF2E34E9A48 2024-08-16 [E]

sec   rsa4096/8C1DBEDB2A60B4EA 2024-03-16 [SC]
      B93F0EF17733EDFF41A673B88C1DBEDB2A60B4EA
uid   [ultimate] Марина <1132236026@pfur.ru>
ssb   rsa4096/0D2212BA2029BA3B 2024-03-16 [E]
```

Рис. 3.11: Вывод списка ключей

Ввожу в терминале команду, с помощью которой копирую сам ключ GPG в буфер обмена, за это отвечает утилита xclip (рис. 3.12).

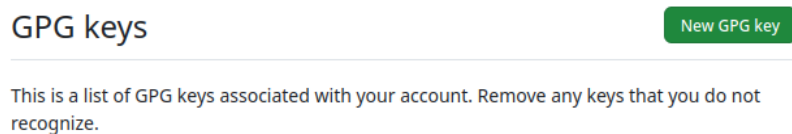


Рис. 3.12: Копирование ключа в буфер обмена

Открываю настройки GitHub, ищу среди них добавление GPG ключа (рис. 3.13).

```
mpzarickaya@mpzarickaya:~$ gpg --armor --export 3D062ADB565EDC58 | xclip -sel clip
mpzarickaya@mpzarickaya:~$
```

Рис. 3.13: Настройки GitHub

Нажимаю на “New GPG key” и вставляю в поле ключ из буфера обмена (рис. 3.14).

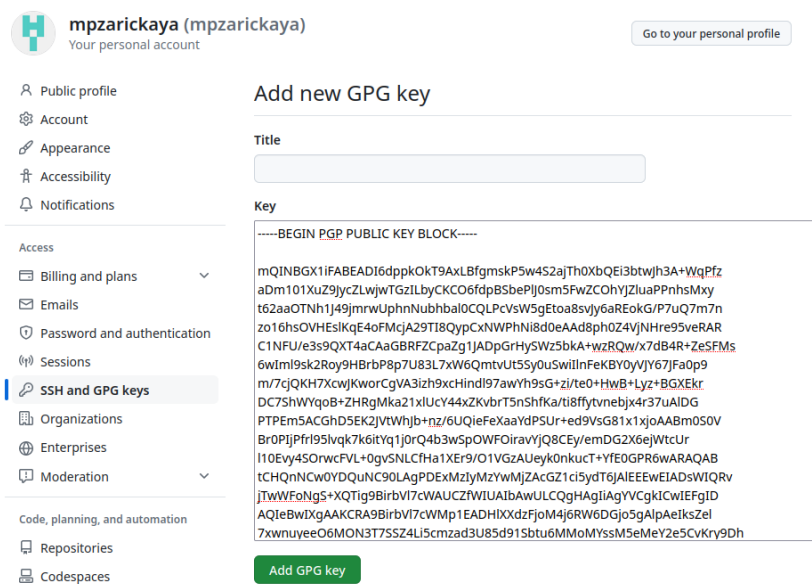


Рис. 3.14: Добавление нового PGP ключа

Я добавила ключ GPG на GitHub (рис. 3.15).

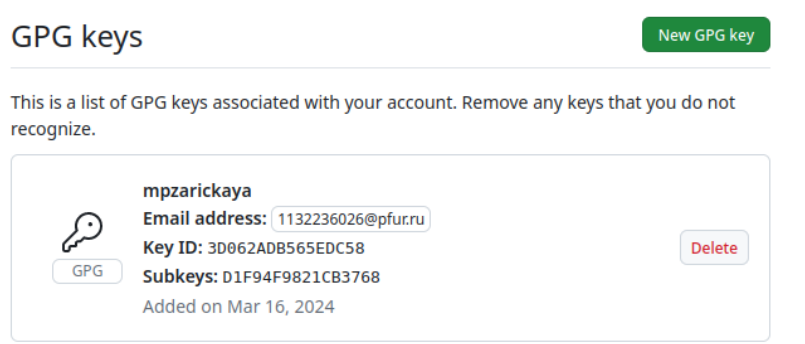


Рис. 3.15: Добавленный ключ GPG

3.7 Настроить подписи Git

Настраиваю автоматические подписи коммитов git: используя введенный ранее email, указываю git использовать его при создании подписей коммитов (рис. 3.16).

```
mpzarickaya@mpzarickaya:~$ git config --global user.signingkey 3D062ADB565EDC58
mpzarickaya@mpzarickaya:~$ git config --global commit.gpgsign true
mpzarickaya@mpzarickaya:~$ git config --global gpg.program $(which gpg2)
mpzarickaya@mpzarickaya:~$
```

Рис. 3.16: Настройка подписей Git

3.8 Настройка gh

Начиная авторизацию в gh, отвечаю на наводящие вопросы от утилиты, в конце выбираю авторизоваться через браузер (рис. 3.17).

```
mpzarickaya@mpzarickaya:~$ gh auth login
? What account do you want to log into? GitHub.com
? What is your preferred protocol for Git operations on this host? HTTPS
? How would you like to authenticate GitHub CLI? Login with a web browser

! First copy your one-time code: 5C8F-5F4E
Press Enter to open github.com in your browser...
```

Рис. 3.17: Авторизация в gh

Завершаю авторизацию на сайте (рис. 3.18).

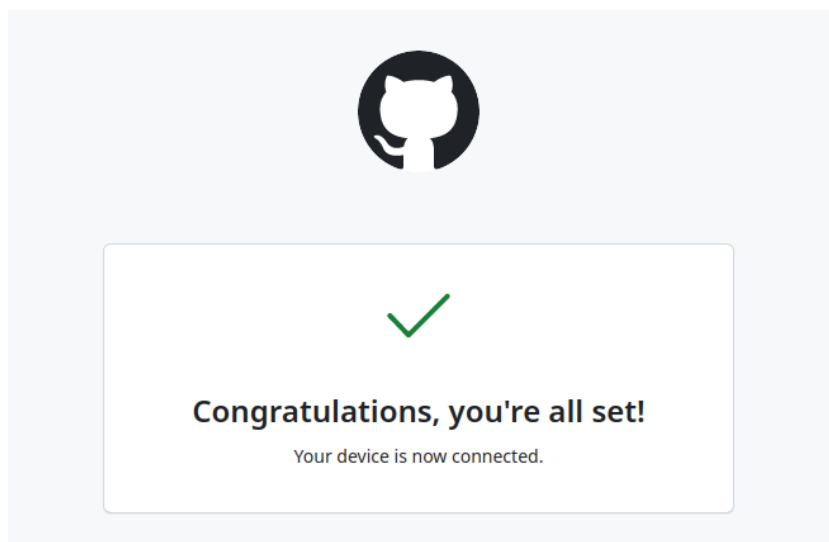


Рис. 3.18: Завершение авторизации через браузер

Вижу сообщение о завершении авторизации под именем mpzarickaya (рис. 3.19).

```
! First copy your one-time code: 5C8F-5F4E
Press Enter to open github.com in your browser...
✓ Authentication complete.
- gh config set -h github.com git_protocol https
✓ Configured git protocol
! Authentication credentials saved in plain text
✓ Logged in as mpzarickaya
! You were already logged in to this account
mpzarickaya@mpzarickaya:~$
```

Рис. 3.19: Завершение авторизации

3.9 Создание репозитория курса на основе шаблона

Сначала создаю директорию с помощью утилиты `mkdir` и флага `-p`, который позволяет установить каталоги на всем указанном пути. После этого с помощью утилиты `cd` перехожу в только что созданную директорию “Операционные системы”. Далее в терминале ввожу команду `gh repo create study_2023-2024_os-intro -template yamadharma/course-directory-student-trmplate -public`, чтобы создать репозиторий на основе шаблона репозитория. После этого клонирую репозиторий к себе в директорию, я указываю ссылку с протоколом `https`, а не `ssh`, потому что при авторизации в `gh` выбрала протокол `https`.

Перехожу в каталог курса с помощью утилиты `cd`, проверяю содержание каталога с помощью утилиты `ls`.

Удаляю лишние файлы с помощью утилиты `rm`, далее создаю необходимые каталоги используя `makefile`.

Добавляю все новые файлы для отправки на сервер (сохраняю добавленные изменения) с помощью команды `git add` и комментирую их с помощью `git commit`.

Отправляю файлы на сервер с помощью `git push`.

4 Выводы

При выполнении данной лабораторной работы я изучила идеологию и применение средств контроля версий, освоила умение по работе с git.

5 Ответы на контрольные вопросы.

1. Системы контроля версий (VCS) - программное обеспечение для облегчения работы с изменяющейся информацией. Они позволяют хранить несколько версий изменяющейся информации, одного и того же документа, может предоставить доступ к более ранним версиям документа. Используется для работы нескольких человек над проектом, позволяет посмотреть, кто и когда внес какое-либо изменение и т. д. VCS применяются для: Хранения полной истории изменений, сохранения причин всех изменений, поиска причин изменений и совершивших изменение, совместной работы над проектами.
2. Хранилище – репозиторий, хранилище версий, в нем хранятся все документы, включая историю их изменения и прочей служебной информацией. commit – отслеживание изменений, сохраняет разницу в изменениях. История – хранит все изменения в проекте и позволяет при необходимости вернуться/обратиться к нужным данным. Рабочая копия – копия проекта, основанная на версии из хранилища, чаще всего последней версии.
3. Централизованные VCS (например: CVS, TFS, AccuRev) – одно основное хранилище всего проекта. Каждый пользователь копирует себе необходимые ему файлы из этого репозитория, изменяет, затем добавляет изменения обратно в хранилище. Децентрализованные VCS (например: Git, Bazaar) – у каждого пользователя свой вариант репозитория (возможно несколько вариантов), есть возможность добавлять и забирать изменения из любого

репозитория. В отличие от классических, в распределенных (децентрализованных) системах контроля версий центральный репозиторий не является обязательным.

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.
7. 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 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.

Список литературы

1. Лабораторная работа № 2 [Электронный ресурс] URL: <https://esystem.rudn.ru/mod/page/view>