

Отчёт по лабораторной работе №2

Отчёт

Кичигина Полина Евгеньевна

Содержание

1	Цель работы	5
2	Задание	6
3	Выполнение лабораторной работы	7
4	Выводы	14
5	Ответы на контрольные вопросы	15

Список иллюстраций

3.1	Устанавливаем git, gh	7
3.2	Базовая настройка git	8
3.3	Настроим верификацию и подписание коммитов git	8
3.4	Генерируем rsa и ed25519	9
3.5	Создаем ключи	10
3.6	Добавляем ключ	11
3.7	Используя введенный email, укажите Git применять его при подписи коммитов	11
3.8	Авторизируемся через браузер	12
3.9	Создаем шаблон	12
3.10	Удаляем лишние файлы и создаем каталоги	13
3.11	Отправка файлов	13

Список таблиц

1 Цель работы

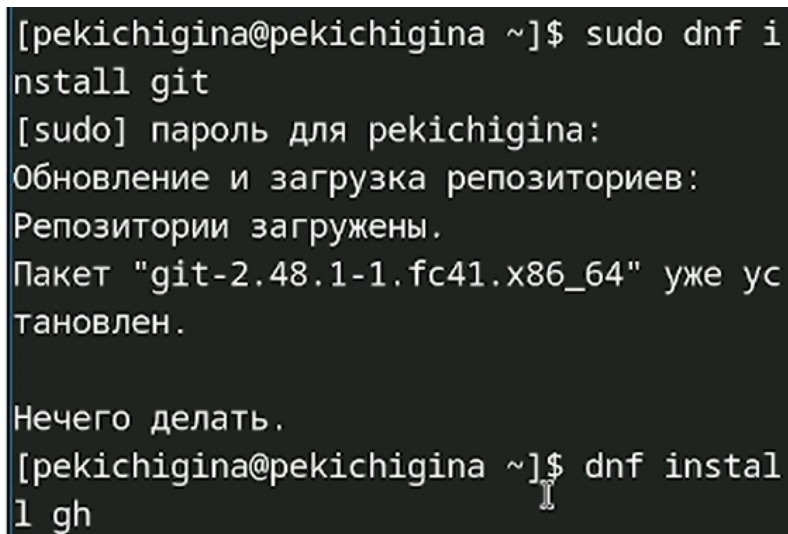
Изучить идеологию и применение средств контроля версий. Освоить умения по работе с git.

2 Задание

Создать базовую конфигурацию для работы с git. Создать ключ SSH. Создать ключ PGP. Настроить подписи git. Зарегистрироваться на Github. Создать локальный каталог для выполнения заданий по предмету.

3 Выполнение лабораторной работы

1. Установка git и gh (рис. 3.1)



```
[pekichigina@pekichigina ~]$ sudo dnf install git
[sudo] пароль для pekichigina:
Обновление и загрузка репозитория:
Репозитории загружены.
Пакет "git-2.48.1-1.fc41.x86_64" уже установлен.

Нечего делать.
[pekichigina@pekichigina ~]$ dnf install gh
```

Рис. 3.1: Устанавливаем git, gh

2. Зададим имя и email владельца репозитория и настроим utf-8 в выводе сообщений git (рис. 3.2)

```
[pekichigina@pekichigina ~]$ git config
--global user.name "Polina Kichigina"
[pekichigina@pekichigina ~]$ git config
--global user.email "polina12071972@gmail.com"
[pekichigina@pekichigina ~]$ git config
--global core.quotepath false
```

Рис. 3.2: Базовая настройка git

Зададим имя начальной ветки(рис. 3.3)

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

Рис. 3.3: Настроим верификацию и подписание коммитов git

3. Создаем ssh ключи(рис. 3.4)


```

home/pekichigina/.ssh/id_rsa
Your public key has been saved in /
/pekichigina/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:12NeK1/1QRljeli9UQ7ZJqCdp/d8Sb1T
pufZWicvxxw pekichigina@pekichigina
The key's randomart image is:
+---[RSA 4096]---+
|           ...B+ |
|          o .B+B |
|         . oo.*+ |
|         . oo.. |
|        S . = oo= |
|         . o +.EB |
|          o +*@ |
|          o.BX |
|          o=+ |
+-----[SHA256]-----+
[pekichigina@pekichigina ~]$ ssh-keygen
-t ed25519

```

Рис. 3.4: Генерируем rsa и ed25519

4. Генерируем ключ pgr(рис. 3.5)

```
[pekichigina@pekichigina ~]$ gpg --
-generate-key
gpg (GnuPG) 2.4.5; Copyright (C) 2019
10 Code GmbH
This is free software: you are free to
change and redistribute it.
There is NO WARRANTY, to the extent per
mitted by law.

gpg: создан каталог '/home/pekichigina/
.gnupg'
Выберите тип ключа:
  (1) RSA and RSA
  (2) DSA and Elgamal
  (3) DSA (sign only)
  (4) RSA (sign only)
  (9) ECC (sign and encrypt) *default*
 (10) ECC (только для подписи)
 (14) Existing key from card

Ваш выбор? 1
```

Рис. 3.5: Создаем ключи

5. Добавление pgr ключа в github

Выводим список ключей и копируем отпечаток приватного ключа(рис. 3.6)

```

[pekichigina@pekichigina ~]$ gpg --
-secret-keys --keyid-format LONG
gpg: проверка таблицы доверия
gpg: marginals needed: 3  completes nee
ded: 1  trust model: pgp
gpg: глубина: 0  достоверных: 1  подп
исанных: 0  доверие: 0-, 0q, 0n, 0m,
0f, 1u
[keyboxd]
-----
sec  rsa4096/9DEF919BEF4139F8 2025-03-
01 [SC]
      79D79A17052585FFFD7A78469DEF919BE
F4139F8
uid                               [ абсолютно ] Polina
Kichigina <polina12071972@gmail.com>
ssb  rsa4096/C279AF9EEFBFAD15 2025-03-
01 [E]

```

Рис. 3.6: Добавляем ключ

6. Настройка автоматических подписей коммитов git (рис. 3.7)

```

[pekichigina@pekichigina ~]$ git config
--global user.signingkey polina1207197
2@gmail.com
[pekichigina@pekichigina ~]$ git config
--global commit.gpgsign true
[pekichigina@pekichigina ~]$ git config
--global gpg.program $(which gpg2)
[pekichigina@pekichigina ~]$ 

```

Рис. 3.7: Используя введённый email, укажите Git применять его при подписи коммитов

7. Настройка gh(рис. 3.8)

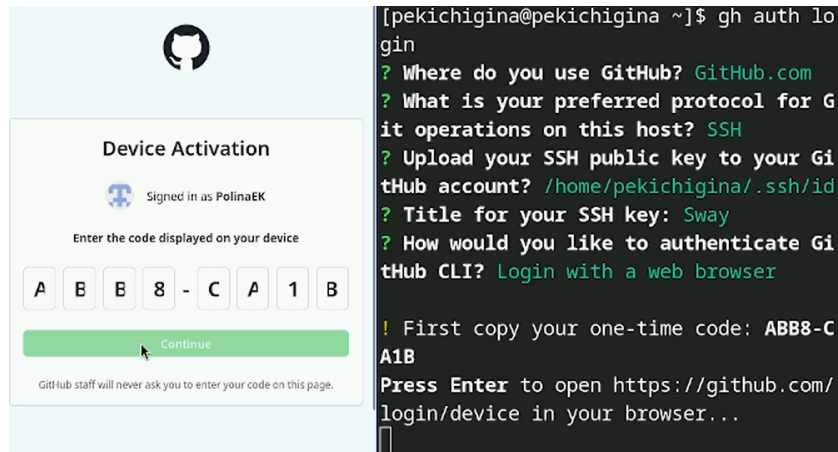


Рис. 3.8: Авторизируемся через браузер

8. Создание репозитория курса на основе шаблона(рис. 3.9)

```
[pekichigina@pekichigina ~]$ mkdir -p ~
/work/study/2024-2025/"Операционные сис
темы"
[pekichigina@pekichigina ~]$ cd ~/work/
study/2024-2025/"Операционные системы"
[pekichigina@pekichigina Операционные с
истемы]$ gh repo create study_2024-2025
_os-intro --template=yamadharma/course-
directory-student-template --public
✓ Created repository PolinaEK/study_202
4-2025_os-intro on GitHub
https://github.com/PolinaEK/study_202
4-2025_os-intro
[pekichigina@pekichigina Операционные с
истемы]$ git clone --recursive git@github
.com:<owner>os-intro
```

Рис. 3.9: Создаем шаблон

9. Настройка каталога курса(рис. 3.10)

```
[pekichigina@pekichigina os-intro]$  
rm package.json  
[pekichigina@pekichigina os-intro]$  
echo os-intro > COURSE  
[pekichigina@pekichigina os-intro]$  
make
```

Рис. 3.10: Удаляем лишние файлы и создаем каталоги

Отправьте файлы на сервер(рис. 3.11)

```
[pekichigina@pekichigina os-intro]$  
git add .  
[pekichigina@pekichigina os-intro]$  
git commit -am 'feat(main): make course structure'
```

Рис. 3.11: Отправка файлов

4 Выводы

Мы изучили идеологию и применение средств контроля версий и освоили умения по работе с git.

5 Ответы на контрольные вопросы

1. Системы контроля версий для отслеживания изменений в файлах, совместной работы и возврата к предыдущим версиям.
2. Хранилище - место для хранения файлов. Commit - запись изменений в хранилище. История - последовательность всех коммитов. Рабочая копия - локальная копия файлов для редактирования.
3. Централизованные - один центральный сервер и все работают с ним. Децентрализованные - у каждого полная копия хранилища.
4. Добавление файлов в рабочую область. Commit изменений.
5. Клонирование хранилища, внесение изменений, commit, push.
6. Управление версиями, совместная работа, отслеживание изменений.
7. git add - добавление в индекс, git push - отправка изменений, git pull - получение изменений, git clone - клонирование.
8. Локальный: git init, git add ., git commit -m "Initial commit". Удаленный: git clone ..., git push.
9. Независимые линии разработки. Нужны для экспериментов, разработки новых функций без влияния на основную копию.
10. Исключение файлов из отслеживания. Нужно для временных файлов, логов, конфиденциальной информации.