

# **Лабораторная работа №2**

**Дисциплина: Операционные системы**

Жибицкая Евгения Дмитриевна

# Содержание

<b>1</b>	<b>Цель работы</b>	<b>5</b>
<b>2</b>	<b>Выполнение лабораторной работы</b>	<b>6</b>
<b>3</b>	<b>Ответы на контрольные вопросы</b>	<b>12</b>
<b>4</b>	<b>Выводы</b>	<b>15</b>

## Список иллюстраций

2.1	Установка git . . . . .	6
2.2	Имя пользователя и почта . . . . .	6
2.3	Настройка параметров . . . . .	7
2.4	Создание ключей . . . . .	7
2.5	Добавление ключей . . . . .	8
2.6	Настройка подписи . . . . .	8
2.7	Авторизация и клонирование . . . . .	9
2.8	Создание нужной структуры . . . . .	10
2.9	Отправка изменений на сервер . . . . .	11

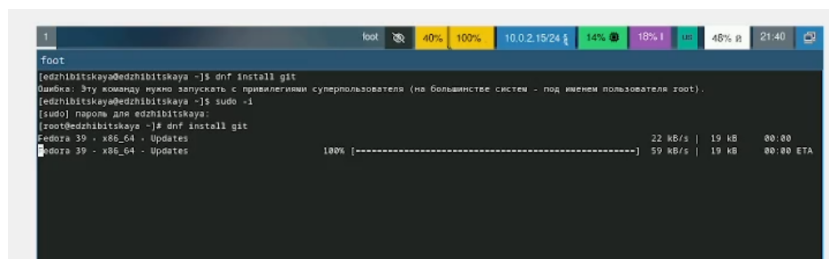
## Список таблиц

# 1 Цель работы

Изучение работы и назначения системы контроля версий git приобретение навыков по работе с ней.

## 2 Выполнение лабораторной работы

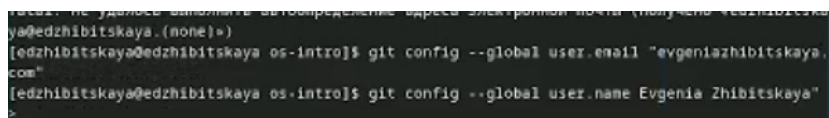
Для начала установим git и fh (рис. 2.1).



```
1
root
[edzhibitskaya@edzhibitskaya ~]$ dnf install git
DnfError: эту команду нужно запустить с привилегиями суперпользователя (на большинстве систем - под именем пользователя root).
[edzhibitskaya@edzhibitskaya ~]$ sudo -i
[sudo] пароль для edzhibitskaya:
[root@edzhibitskaya ~]# dnf install git
Fedora 39 - x86_64 - Updates                22 kB/s | 19 kB | 00:00
Fedora 39 - x86_64 - Updates                59 kB/s | 19 kB | 00:00 ETA
100% [=====]
```

Рис. 2.1: Установка git

После этого необходимо указать имя и почту (рис. 2.2), также настроить utf-8 и параметры autocrlf, safecrlf (рис. 2.3).



```
ya@edzhibitskaya.(none)*)
[edzhibitskaya@edzhibitskaya os-intro]$ git config --global user.email "evgeniazhibitskaya.com"
[edzhibitskaya@edzhibitskaya os-intro]$ git config --global user.name Evgenia Zhibitskaya
```

Рис. 2.2: Имя пользователя и почта

```
foot
[edzhibitskaya@edzhibitskaya ~]$ sudo -i
[sudo] пароль для edzhibitskaya:
Попробуйте ещё раз.
[sudo] пароль для edzhibitskaya:
[root@edzhibitskaya ~]# git config --global core.quotepath false
[root@edzhibitskaya ~]# git config --global init.defaultBranch master
[root@edzhibitskaya ~]# git config --global core.autocrlf input
[root@edzhibitskaya ~]#
```

Рис. 2.3: Настройка параметров

Далее создаем ключ ssh и ключ pgp (рис. 2.4).

```
foot
[edzhibitskaya@edzhibitskaya ~]$ gpg --list-secret-keys --keyid-format LONG
gpg: создан кавалор "/home/edzhibitskaya/.gnupg"
gpg: /home/edzhibitskaya/.gnupg/trustdb.gpg: создана таблица доверия
[edzhibitskaya@edzhibitskaya ~]$ ssh-keygen -t rsa -b 4096
Generating public/private rsa key pair.
Enter file in which to save the key (/home/edzhibitskaya/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/edzhibitskaya/.ssh/id_rsa
Your public key has been saved in /home/edzhibitskaya/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:nHt5Z79aLY3hT1u0g5yC/IXaM0xIM3wLuz/qjs5g51Y edzhibitskaya@edzhibitskaya
The key's randomart image is:
+----[RSA 4096]----+
|
| o . .
| o S . . . *
| . o.Ooo ooO |
| .E oo B -oo *oo|
| .+..+ ..oo+ +..|
| .o. .o**o+
+-----[SHA256]-----+
[edzhibitskaya@edzhibitskaya ~]$ ssh-keygen -t ed25519
Generating public/private ed25519 key pair.
Enter file in which to save the key (/home/edzhibitskaya/.ssh/id_ed25519):
/home/edzhibitskaya/.ssh/id_ed25519 already exists.
Overwrite (y/n)? y
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/edzhibitskaya/.ssh/id_ed25519
Your public key has been saved in /home/edzhibitskaya/.ssh/id_ed25519.pub
The key fingerprint is:
SHA256:T55aq73aA0z0Fu058sy5ned7w5BgM6cvVWw2V9f0e/N edzhibitskaya@edzhibitskaya
The key's randomart image is:
+--[ED25519 256]--+
|
| . . . . *
| - + + . *
| ..B . o . |
| oo.. . . |
| =So.. oo . |
| B=+o o+ |
| . **o E |
+-----
```

Рис. 2.4: Создание ключей

На платформе github учетная запись была создана ранее, поэтому просто ав-

торизуемся и добавим созданные ключи туда, скопировав их отпечатки (рис. 2.5).

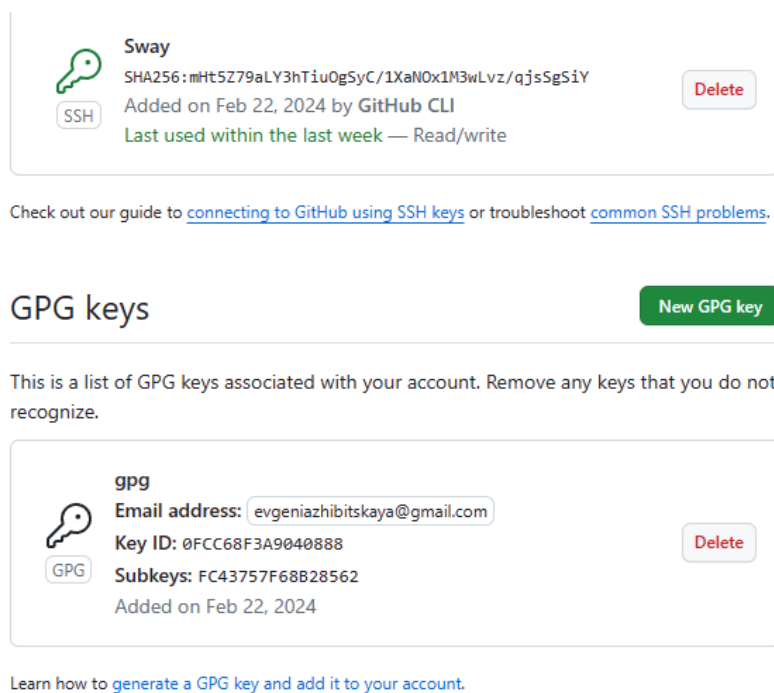


Рис. 2.5: Добавление ключей

Следующим шагом настроим автоматическую подпись коммитов, введя следующие команды (рис. 2.6).

```
edzhibitskaya@edzhibitskaya ~$ git config --global user.signingkey evgeniazhibitskaya@gmail.com
edzhibitskaya@edzhibitskaya ~$ git config --global commit.gpgsign true
edzhibitskaya@edzhibitskaya ~$ git config --global gpg.program $(which gpg2)
edzhibitskaya@edzhibitskaya ~$ gh auth login
```

Рис. 2.6: Настройка подписи

Наконец авторизируемся с помощью команды `gh login auth` и, создав и перейдя в нужные каталоги сначала создадим репозиторий на основе шаблона, а затем клонируем его себе (рис. 2.7).



```

edzhibitskaya@edzhibitskaya:~$ git config --global gpg.program gpg
edzhibitskaya@edzhibitskaya:~$ gh auth login
? What account do you want to log into? github.com
? What is your preferred protocol for Git operations on this host? SSH
? Upload your SSH public key to your GitHub account? /home/edzhibitskaya/.ssh/id_rsa.pub
? Title for your SSH key: Sshkey
? How would you like to authenticate GitHub CLI? Login with a web browser

First copy your one-time code: 5161-6A83
Press Enter to open github.com in your browser...
Authentication complete.
gh config set -h github.com git_protocol ssh
Configured git protocol
Uploaded the SSH key to your GitHub account: /home/edzhibitskaya/.ssh/id_rsa.pub
Logged in as JaneZhibit
You were already logged in to this account
edzhibitskaya@edzhibitskaya:~$ cd ~/work/study/2022-2023/'Операционные системы'
bash: cd: /home/edzhibitskaya/work/study/2022-2023/Операционные системы: Нет такого файла или каталога
edzhibitskaya@edzhibitskaya:~$ cd ~/work/study/2023.2024/Операционные системы/
edzhibitskaya@edzhibitskaya:~/work/study/2023.2024/Операционные системы$ git clone --recursive git@github.com:owner/study_2022-2023_os-intro.git os-intro
bash: owner: Нет такого файла или каталога
edzhibitskaya@edzhibitskaya:~/work/study/2023.2024/Операционные системы$ git clone --recursive git@github.com:JaneZhibit/study_2023-2024_os-intro.git os-intro
Клонирование в «os-intro»...
remote: Enumerating objects: 32, done.
remote: Counting objects: 100% (32/32), done.

```

Рис. 2.7: Авторизация и клонирование

Также, перейдя в нужный каталог, удалим лишние файлы и создадим необходимые каталоги (рис. 2.8).

```
foot
Удостоверьтесь, что у вас есть необходимые права доступа
и репозиторий существует.
[root@edzhbitskaya: Операционные системы]#
выход
[edzhbitskaya@edzhbitskaya os-intro]$ echo os-intro > COURSE
[edzhbitskaya@edzhbitskaya os-intro]$ make
Usage:
  make <target>

Targets:
  list           List of courses
  prepare        Generate directories structure
  submodule       Update submodules

[edzhbitskaya@edzhbitskaya os-intro]$ make list
      net-admin  Администрирование локальных сетей
      net-os-admin Администрирование сетевых подсистем
      arch-pc     Архитектура ЭВМ
      sciprog-intro Введение в научное программирование
      infosec     Информационная безопасность
      computer-practice Компьютерный практикум по статистическому анализу д
анных
      mathsec     Математические основы защиты информации и информаци
онной безопасности
      mathmod      Математическое моделирование
      simulation-networks Моделирование сетей передачи данных
      sciprog      Научное программирование
      os-intro     Операционные системы
[edzhbitskaya@edzhbitskaya os-intro]$ make prepare
[edzhbitskaya@edzhbitskaya os-intro]$ make submodule
git submodule update --init --recursive
git submodule foreach 'git fetch origin; git checkout $(git rev-parse --abbrev-ref HEAD); git reset --hard origin/$(git rev-parse --abbrev-ref HEAD); git submodule update --recursive; git clean -dfx'
Entering 'template/presentation'
Указатель HEAD сейчас на коммите 40a1761 Merge branch 'release/1.0.3'
Entering 'template/report'
Указатель HEAD сейчас на коммите 7c31ab8 Merge branch 'release/1.0.4'
[edzhbitskaya@edzhbitskaya os-intro]$ ls
CHANGELOG.md  labs      prepare      README.en.md  template
config        LICENSE   presentation README.git-flow.md
COURSE        Makefile  project-personal README.md
[edzhbitskaya@edzhbitskaya os-intro]$ git add .
[edzhbitskaya@edzhbitskaya os-intro]$ git commit -am 'feat(main'
```

Рис. 2.8: Создание нужной структуры

Отправим все изменения на github (рис. ??).

```
xnos/pandocattributes.py
create mode 100644 project-personal/stage6/report/report.md
[edzhbitskaya@edzhbitskaya os-intro]$ git push
Перечисление объектов: 40, готово.
Подсчет объектов: 100% (40/40), готово.
При скатии изменений используется до 2 потоков
Сжатие объектов: 100% (30/30), готово.
Запись объектов: 100% (38/38), 342.11 КиБ | 2.74 МБ/с, готово.
Всего 38 (изменений 4), повторно использовано 0 (изменений 0), повторно
использовано пакетов 0
remote: Resolving deltas: 100% (4/4), completed with 1 local object.
To github.com:JaneZhibit/study_2023-2024_os-intro.git
 f029ada..1549115 master -> master
[edzhbitskaya@edzhbitskaya os-intro]$
```

Рис. 2.9: Отправка изменений на сервер

### 3 Ответы на контрольные вопросы

1. Что такое системы контроля версий (VCS) и для решения каких задач они предназначаются?
  - Они применяются при работе нескольких человек с одним проектом. При внесении изменений позволяют фиксировать, совмещать и возвращать изменения разных людей
2. Объясните следующие понятия VCS и их отношения: хранилище, commit, история, рабочая копия. Хранилище - место, где находятся данные(файлы, коды и тд) Commit - команда, для сохранения изменений История - информация о предыдущих изменениях Рабочая копия - одна из версий проекта, с которой ведется работа(= текущая/основная)
3. Что представляют собой и чем отличаются централизованные и децентрализованные VCS? Приведите примеры VCS каждого вида.
  - Централизованные системы предполагают наличие единого репозитория для хранения данных(CVC, Subversion)
  - В децентрализованных системах центральный репозиторий не обязателен(Git, Bazaar)
4. Опишите действия с VCS при единоличной работе с хранилищем. Работа происходит на своем компьютере, сначала обновляются данные, в конце они размещаются в центральном репозитории

5. Опишите порядок работы с общим хранилищем VCS. Для идентификации на сервере необходимы ключи и затем создание репозитория, только затем можно работать на локальной машине. Также в конце изменения добавляются на сервер
6. Каковы основные задачи, решаемые инструментальным средством git?
  - хранение информации о всех изменениях
  - обеспечение удобства командной работы
7. Назовите и дайте краткую характеристику командам 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 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 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. Приведите примеры использования при работе с локальным и удалённым репозиториями. локальный репозиторий - работа со своими файлами удаленный репозиторий - совместная работа, общий проект
9. Что такое и зачем могут быть нужны ветви (branches)? Ветви это пути к отдельным состояниям (отделы проекта),они дают возможность вносить изменения только в часть проекта и не трогать все вышестоящее
10. Как и зачем можно игнорировать некоторые файлы при commit? Игнорировать файлы можно с помощью .gitignore. Нужно это, например, при наличии ненужных(лишних или созданных автоматически) файлов.

## 4 Выводы

В ходе работы была освоена работа с системой контроля версий, был установлен git, проведена авторизация, заданы базовые настройки, создались ключи, клонировался репозиторий и так далее.