

# **Отчёт по лабораторной работе №2**

Уткина Алина Дмитриевна

# Содержание

<b>1</b>	<b>Цель работы</b>	<b>4</b>
<b>2</b>	<b>Задание</b>	<b>5</b>
<b>3</b>	<b>Выполнение лабораторной работы</b>	<b>6</b>
3.1	Установка программного обеспечения . . . . .	6
3.2	Базовая настройка git . . . . .	6
3.3	Создание ключей ssh . . . . .	7
3.4	Создание ключей pgr . . . . .	8
3.5	Настройка GitHub . . . . .	8
3.6	Добавление PGP ключа в GitHub . . . . .	9
3.7	Настройка автоматических подписей коммитов git . . . . .	9
3.8	Настройка gh . . . . .	10
3.9	Шаблон для рабочего пространства . . . . .	10
3.10	Настройка каталога курса . . . . .	11
3.11	Ответы на контрольные вопросы . . . . .	11
<b>4</b>	<b>Выводы</b>	<b>15</b>

## Список иллюстраций

3.1	Установка git . . . . .	6
3.2	Установка gh . . . . .	6
3.3	Настройка git . . . . .	7
3.4	Создание ключа по алгоритму rsa . . . . .	7
3.5	Создание ключа по алгоритму ed25519 . . . . .	8
3.6	Генерация ключа pgr . . . . .	8
3.7	Список ключей . . . . .	9
3.8	Добавление GPG ключа на GitHub . . . . .	9
3.9	Настройка подписей коммитов . . . . .	10
3.10	Авторизация . . . . .	10
3.11	Создание репозитория курса . . . . .	10
3.12	Настройка каталога курса . . . . .	11
3.13	Примеры команд . . . . .	13

# 1 Цель работы

Целью данной работы является изучение идеологии и применение средств контроля версий, освоение 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

### 3.1 Установка программного обеспечения

Установим git (рис. 3.1):

```
[adutkina@fedora ~]$ sudo -i
[sudo] пароль для adutkina:
[root@fedora ~]# dnf install git
Последняя проверка окончания срока действия метаданных: 2:35:19 назад,
Пт 17 фев 2023 18:25:44.
Пакет git-2.39.1-1.fc36.x86_64 уже установлен.
Зависимости разрешены.
Отсутствуют действия для выполнения.
Выполнено!
```

Рис. 3.1: Установка git

Установим gh (рис. 3.2):

```
Установлен:
gh-2.22.1-1.fc36.x86_64

Выполнено!
[root@fedora ~]#
```

Рис. 3.2: Установка gh

### 3.2 Базовая настройка git

Зададим имя и email владельца репозитория, настроим utf-8 в выводе сообщений git, настроим верификацию и подписание коммитов git, зададим имя начальной ветки (будем называть её master) и установим параметры autocrlf и safecrlf (рис. 3.3).

```
[adutkina@fedora ~]$ git config --global user.name "Alina Utkina"
[adutkina@fedora ~]$ git config --global user.email "arccn2004@gmail.com"
[adutkina@fedora ~]$ git config --global core.quotePath false
[adutkina@fedora ~]$ git config --global init.defaultBranch master
[adutkina@fedora ~]$ git config --global core.autocrlf input
[adutkina@fedora ~]$ git config --global core.safecrlf warn
```

Рис. 3.3: Настройка git

### 3.3 Создание ключей ssh

Создадим ключи ssh по алгоритму rsa с ключём размером 4096 бит (рис. 3.4) и алгоритму ed25519 (рис. 3.5).

```
[adutkina@fedora ~]$ ssh-keygen -t rsa -b 4096
Generating public/private rsa key pair.
Enter file in which to save the key (/home/adutkina/.ssh/id_rsa):
/home/adutkina/.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/adutkina/.ssh/id_rsa
Your public key has been saved in /home/adutkina/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:JWoHe/TLYExMpuqz73G+FyIVyczdifpc9b8mj/TF48c adutkina@fedora
The key's randomart image is:
+---[RSA 4096]-----+
|      +oo o .      |
|      =* o o .      |
|      o =o. . .      |
|      . Bo+ . .      |
|      . +.So.. .      |
|      . ..+.o+. .      |
|      o ....o. . o+      |
|      o + . ..+oE      |
|      .oo oo .++      |
|      +-----+
+---[SHA256]-----+
[adutkina@fedora ~]$
```

Рис. 3.4: Создание ключа по алгоритму rsa

```
[adutkina@fedora ~]$ ssh-keygen -t ed25519
Generating public/private ed25519 key pair.
Enter file in which to save the key (/home/adutkina/.ssh/id_ed25519):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/adutkina/.ssh/id_ed25519
Your public key has been saved in /home/adutkina/.ssh/id_ed25519.pub
The key fingerprint is:
SHA256:qMLM59x0u/9Eom7hyKImd6F4FYpYUxeG5efGIMnbUzo adutkina@fedora
The key's randomart image is:
+--[ED25519 256]--+
|      o+.      |
|     .o+.     |
|    .+.o o    |
|   o .+ 0     |
|  .o oo.E S . |
| o .+o* =.. o |
| . o=ooo+. .  |
| o = o o+o. . |
| =.o . . =o.. |
+----[SHA256]-----+
[adutkina@fedora ~]$
```

Рис. 3.5: Создание ключа по алгоритму ed25519

### 3.4 Создание ключей pgr

Генерируем ключ командой “gpg –full-generate-key”, из предложенных опций выбираем: тип RSA and RSA, размер 4096 и срок действия 0 (срок действия не истекает никогда). Зададим личную информацию, которая сохранится в ключе: имя, адрес электронной почты, используемый на GitHub (рис. 3.6)

```
pub  rsa4096 2023-02-17 [SC]
    126076ACACB2CD4EF865C3F7C7734EE8C25CFF39
uid                               Alina <arccn2004@gmail.com>
sub  rsa4096 2023-02-17 [E]
[adutkina@fedora ~]$
```

Рис. 3.6: Генерация ключа pgr

### 3.5 Настройка GitHub

Заходим в созданную учетную запись и проверяем основные данные



## 3.6 Добавление PGP ключа в GitHub

Выводим список ключей и копируем отпечаток приватного ключа (рис. 3.7).

```
sec  rsa4096/C7734EE8C25CFF39 2023-02-17 [SC]  
126076ACACB2CD4EF865C3F7C7734EE8C25CFF39  
uid  [ абсолютно ] Alina <arccn2004@gmail.com>  
ssb  rsa4096/FA91F5D57E8AA9EE 2023-02-17 [E]
```

Рис. 3.7: Список ключей

Отпечаток ключа — это последовательность байтов, используемая для идентификации более длинного, по сравнению с самим отпечатком ключа. Формат строки:

- sec Алгоритм/Отпечаток ключа, Дата создания [Флаги] [Годен\_до]
- ID\_ключа

Скопируем сгенерированный PGP ключ в буфер обмена: `gpg --armor --export C7734EE8C25CFF39 | xclip -sel clip`. Перейдем в настройки GitHub и вставим полученный ключ в поле ввода (рис. 3.8).

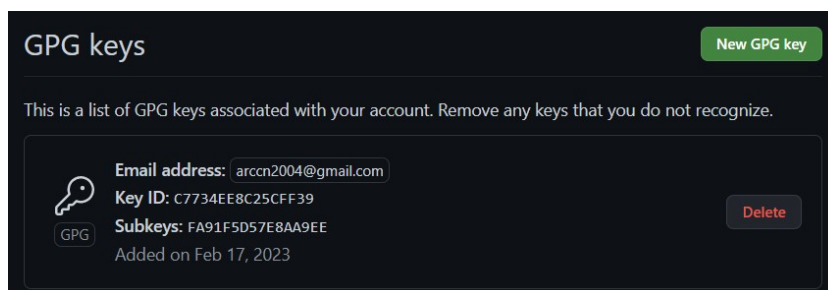


Рис. 3.8: Добавление GPG ключа на GitHub

## 3.7 Настройка автоматических подписей коммитов git

Используя введенный email, укажем Git применять его при подписи коммитов (рис. 3.9).

```
[adutkina@fedora ~]$ git config --global user.signingkey C7734EE8C25CFF39
[adutkina@fedora ~]$ git config --global commit.gpgsign true
[adutkina@fedora ~]$ git config --global gpg.program $(which gpg2)
```

Рис. 3.9: Настройка подписей коммитов

## 3.8 Настройка gh

Для начала войдем в аккаунт, ответив на несколько наводящих вопросов (рис. 3.10).

```
[adutkina@fedora Операционные системы]$ gh auth login
? What account do you want to log into? GitHub.com
? What is your preferred protocol for Git operations? SSH
? Upload your SSH public key to your GitHub account? /home/adutkina/.ssh/id_rsa.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: A1A1-577A
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/adutkina/.ssh/id_rsa.pub
✓ Logged in as UAlina
```

Рис. 3.10: Авторизация

## 3.9 Шаблон для рабочего пространства

Создадим репозиторий курса на основе шаблона, для этого создадим каталог `~/work/study/2022-2023/“Операционные системы”`, скачаем шаблон и клонируем его в каталог (рис. 3.11).

```
[adutkina@fedora Операционные системы]$ gh repo create study_2022-2023_os-intro --template=yamadharm/course-directory-student-template --public
✓ Created repository UAlina/study_2022-2023_os-intro on GitHub
[adutkina@fedora Операционные системы]$ git clone --recursive git@github.com:UAlina/study_2022-2023_os-intro.git os-intro
Клонирование в «os-intro»...
```

Рис. 3.11: Создание репозитория курса

## 3.10 Настройка каталога курса

Перейдем в каталог курса, удалим лишние файлы, создадим необходимые каталоги и отправим файлы на сервер (рис. 3.12)

```
[adutkina@fedora os-intro]$ rm package.json
[adutkina@fedora os-intro]$ echo os-intro > COURSE
[adutkina@fedora os-intro]$ make
[adutkina@fedora os-intro]$ ls
CHANGELOG.md  labs      prepare      README.en.md  template
config        LICENSE   presentation  README.git-flow.md
COURSE        Makefile  project-personal  README.md
[adutkina@fedora os-intro]$ git add .
[adutkina@fedora os-intro]$ git commit -am 'feat(main): make course struture'
[master 067bcf5] feat(main): make course struture
361 files changed, 100327 insertions(+), 14 deletions(-)
create mode 100644 labs/README.md
create mode 100644 labs/README.ru.md
```

Рис. 3.12: Настройка каталога курса

## 3.11 Ответы на контрольные вопросы

1. Системы контроля версий (VCS) разработаны специально для того, чтобы максимально упростить и упорядочить работу над проектом (вне зависимости от того, сколько человек в этом участвуют). СКВ дает возможность видеть, кто, когда и какие изменения вносил; позволяет формировать новые ветви проекта, объединять уже имеющиеся; настраивать контроль доступа к проекту; осуществлять откат до предыдущих версий.
2. Основные понятия:
  - Хранилище (repository, сокр. репо), или репозиторий, — место хранения всех версий и служебной информации;
  - Коммит (commit) — 1) синоним версии; 2) создание новой версии («сделать коммит», «закоммитить»);
  - История разработки — совокупность всех версий файлов, над которыми ведется работа. Историей разработки в данном случае будет список изменений: создание файла, добавление изначального текста, исправление

опечатки, добавление нового текста, объединение двух версий файла (при выполнении слияния);

- Рабочая копия (working copy или working tree) — текущее состояние файлов проекта, основанное на версии из хранилища (обычно на последней).

### 3. Централизованные и децентрализованные VCS:

- Централизованные VCS - одно основное хранилище всего проекта, где каждый пользователь копирует себе необходимые ему файлы из этого репозитория, изменяет и, затем, добавляет свои изменения обратно. Например Subversion, CVS, TFS, VAULT, AccuRev;
- Децентрализованные VCS - у каждого пользователя свой вариант (возможно не один) репозитория, присутствует возможность добавлять и забирать изменения из любого репозитория. Например Git, Mercurial, Bazaar.

### 4. Единоличная работа с хранилищем:

- работа в локальном репозитории;
- сохранение изменений и загрузка на серверов.

### 5. Работа с общим хранилищем VCS:

- проверка обновлений;
- загрузка обновлений (при наличии);
- работа в локальном репозитории;
- создаются ветвления, если несколько пользователей работают над одним и тем же файлом/документом;
- по результатам различных версий могут происходить слияния в одну ветвь.

### 6. Основные задачи, решаемые инструментальным средством git:

- хранить информацию о всех изменениях в коде;
- обеспечение удобства командной работы над кодом.

## 7. Примеры команд git:

- git pull - получение обновлений (изменений) текущего дерева из центрального репозитория;
- git push - отправка всех произведённых изменений локального дерева в центральный репозиторий;
- git status - просмотр списка изменённых файлов в текущей директории;
- git add - добавить все изменённые и/или созданные файлы и/или каталоги;
- git commit -am 'Описание коммита' - сохранить все добавленные изменения и все изменённые файлы.

## 8. Примеры команд для работы с локальным и удалённым репозиториями (рис. 3.13)

```
[adutkina@fedora image]$ git add .
[adutkina@fedora image]$ git commit -am 'feat(main): add images lab-02'
[master 58ad62f] feat(main): add images lab-02
14 files changed, 124 insertions(+), 32 deletions(-)
create mode 100644 labs/lab02/report/image/1.jpg
create mode 100644 labs/lab02/report/image/10.jpg
create mode 100644 labs/lab02/report/image/11.jpg
create mode 100644 labs/lab02/report/image/12.jpg
create mode 100644 labs/lab02/report/image/2.jpg
create mode 100644 labs/lab02/report/image/3.jpg
create mode 100644 labs/lab02/report/image/4.jpg
create mode 100644 labs/lab02/report/image/5.jpg
create mode 100644 labs/lab02/report/image/6.jpg
create mode 100644 labs/lab02/report/image/7.jpg
create mode 100644 labs/lab02/report/image/8.jpg
create mode 100644 labs/lab02/report/image/9.jpg
delete mode 100644 labs/lab02/report/image/placeimg_800_600_tech.jpg
[adutkina@fedora image]$ git push
Перечисление объектов: 25, готово.
Подсчет объектов: 100% (25/25), готово.
Сжатие объектов: 100% (19/19), готово.
Запись объектов: 100% (19/19), 596.73 Киб | 5.47 Миб/с, готово.
```

Рис. 3.13: Примеры команд

9. Ветка (англ. branch) — это последовательность коммитов, в которой ведётся параллельная разработка какого-либо функционала. Основная ветка – master. Ветки нужны, чтобы несколько программистов могли вести работу над одним и тем же проектом или даже файлом одновременно, при этом не мешая друг другу. Кроме того, ветки используются для тестирования экспериментальных функций: чтобы не повредить основному проекту, создается новая ветка специально для экспериментов.

10. Для игнорирования некоторых файлов можно создать файл `.gitignore` в корневом каталоге репозитория, чтобы сообщить Git, какие файлы и каталоги следует игнорировать при фиксации. Иногда имеется группа файлов, которые не нужно автоматически добавлять в репозиторий. К таким файлам обычно относятся автоматически генерируемые файлы (различные логи, результаты сборки программ и т. п.).

## 4 Выводы

В ходе данной работы были изучены идеологии и применение средств контроля версий, освоены умения по работе с git.