

# **Отчёт по лабораторной работе №2**

**Специальность: архитектура компьютеров**

Ерфан Хосейнабади

# Содержание

|          |  |           |
|----------|--|-----------|
| <b>1</b> | <b>Цель работы</b>   | <b>5</b>  |
| <b>2</b> | <b>Задание</b>   | <b>6</b>  |
| <b>3</b> | <b>Теоретические сведения</b>                                      | <b>7</b>  |
| <b>4</b> | <b>Задание для самостоятельной работы</b>                          | <b>9</b>  |
| 4.1      | 1. Создание базовой конфигурации для работы с <b>Git</b> . . . . . | 9         |
| 4.2      | 2. Создание ключа <b>SSH</b> . . . . .                             | 9         |
| 4.3      | 3. Создание ключа <b>PGP</b> . . . . .                             | 11        |
| 4.4      | 4. Настройка подписей <b>Git</b> . . . . .                         | 13        |
| 4.5      | 5. Регистрация на <b>GitHub</b> . . . . .                          | 14        |
| 4.6      | 6. Создание локального каталога для выполнения заданий . . . . .   | 15        |
| 4.7      | Использование репозитория . . . . .                                | 15        |
|          | 4.7.1 Удаление лишних файлов . . . . .                             | 16        |
|          | 4.7.2 Создание необходимых каталогов . . . . .                     | 16        |
| 4.8      | Выводы . . . . .   | 17        |
| <b>5</b> | <b>Ответы на контрольные вопросы.</b>                              | <b>18</b> |

## **Список иллюстраций**

## **Список таблиц**

# 1 Цель работы

- Изучить идеологию и применение средств контроля версий.
- Освоить 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 Теоретические сведения

Системы контроля версий (Version Control System, VCS) применяются при работе нескольких человек над одним проектом. Обычно основное дерево проекта хранится в локальном или удалённом репозитории, к которому настроен доступ для участников проекта. При внесении изменений в содержание проекта система контроля версий позволяет их фиксировать, совмещать изменения, произведённые разными участниками проекта, производить откат к любой более ранней версии проекта, если это требуется. В классических системах контроля версий используется централизованная модель, предполагающая наличие единого репозитория для хранения файлов. Выполнение большинства функций по управлению версиями осуществляется специальным сервером. Участник проекта (пользователь) перед началом работы посредством определённых команд получает нужную ему версию файлов. После внесения изменений, пользователь размещает новую версию в хранилище. При этом предыдущие версии не удаляются из центрального хранилища и к ним можно вернуться в любой момент. Сервер может сохранять не полную версию изменённых файлов, а производить так называемую дельта-компрессию — сохранять только изменения между последовательными версиями, что позволяет уменьшить объём хранимых данных. Системы контроля версий поддерживают возможность отслеживания и разрешения конфликтов, которые могут возникнуть при работе нескольких человек над одним файлом. Можно объединить (слить) изменения, сделанные разными участниками (автоматически или вручную), вручную выбрать нужную версию, отменить изменения вовсе или заблокировать файлы для изменения. В зави-

симости от настроек блокировка не позволяет другим пользователям получить рабочую копию или препятствует изменению рабочей копии файла средствами файловой системы ОС, обеспечивая таким образом, привилегированный доступ только одному пользователю, работающему с файлом. Системы контроля версий также могут обеспечивать дополнительные, более гибкие функциональные возможности. Например, они могут поддерживать работу с несколькими версиями одного файла, сохраняя общую историю изменений до точки ветвления версий и собственные истории изменений каждой ветви. Кроме того, обычно доступна информация о том, кто из участников, когда и какие изменения вносил. Обычно такого рода информация хранится в журнале изменений, доступ к которому можно ограничить. В отличие от классических, в распределённых системах контроля версий центральный репозиторий не является обязательным. Среди классических VCS наиболее известны CVS, Subversion, а среди распределённых — Git, Bazaar, Mercurial. Принципы их работы схожи, отличаются они в основном синтаксисом используемых в работе команд

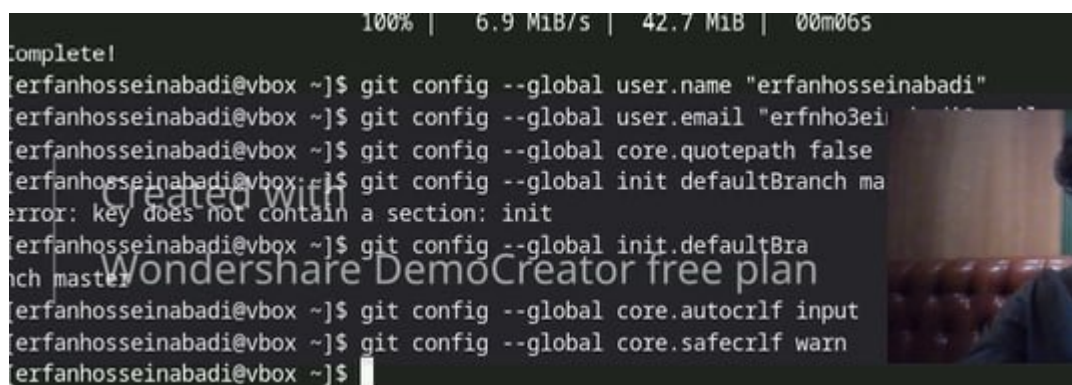


## 4 Задание для самостоятельной работы

### 4.1 1. Создание базовой конфигурации для работы с Git

Для начала необходимо выполнить базовую настройку **Git**, задав имя и email владельца репозитория. Используйте следующие команды:

```
git config --global user.name "Ваше Имя"
git config --global user.email "ваш_email@example.com"
```



{#fig:001

width:70%}

### 4.2 2. Создание ключа SSH

Следующим шагом является создание ключа **SSH** по алгоритму RSA с размером ключа **4096** бит. Введите команду:

```
ssh-keygen -t rsa -b 4096 -C "ваш_email@example.com"
```

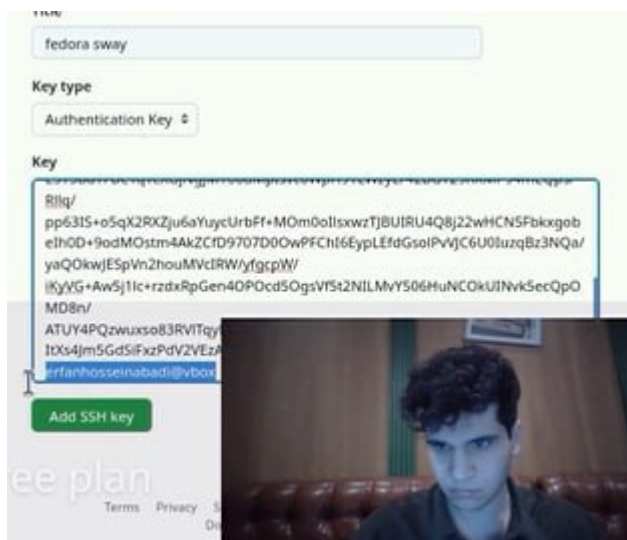
При выполнении этой команды будет предложено указать место для сохранения ключа. Нажмите Enter, чтобы использовать значение по умолчанию.

```
[erfanhosseinabadi@vbox ~]$ ssh-keygen -t rsa -b 4096
bash: ssh_keygen: command not found
[erfanhosseinabadi@vbox ~]$ ssh-keygen -t rsa -b 4096
Generating public/private rsa key pair.
Enter file in which to save the key (/home/erfanhosseinabadi/.ssh/id_rsa):
Created directory '/home/erfanhosseinabadi/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/erfanhosseinabadi/.ssh/id_rsa
Your public key has been saved in /home/erfanhosseinabadi/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:Bqts4jq+BzYwIj3dzd6ITQhGq2HMDcsrOsHXWJBKtCU erfanhosseinabadi@vbox
The key's randomart image is:
+---[RSA 4096]-----+
|.E.+o|
|+=+=.|
|.oOoo+.+|
|*oo+= oo+|
|*oo+ ..5p|
|.+. . . .+|
|+ + +|
|.o +|
|o=+|
+---[SHA256]-----+
[erfanhosseinabadi@vbox ~]$
```

{#fig:002

width:70%}

После этого вы можете скопировать сгенерированный SSH ключ из файла и вставить его в **GitHub** в разделе настроек SSH.



{#fig:003 width:70%}

### 4.3 3. Создание ключа PGP

Для создания ключа PGP выполните команду:

```
gpg --full-generate-key
```

Выберите параметры, указанные на изображениях, и завершите процесс генерации ключа.

```

root
[erfanhosseinabadi@vbox ~]$ gpg --full-generate-key
gpg (GnuPG) 2.4.5; 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.

gpg: directory '/home/erfanhosseinabadi/.gnupg' created
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

```



Created with  
Wondershare DemoCreator free plan

{#fig:004

```

We need to generate a lot of random bytes. It is a good idea to perform
some other action (type on the keyboard, move the mouse, utilize the
disks) during the prime generation; this gives the random number
generator a better chance to gain enough entropy.
gpg: /home/erfanhosseinabadi/.gnupg/trustdb.gpg: trustdb created
gpg: directory '/home/erfanhosseinabadi/.gnupg/openpgp-revocs.d' created
gpg: revocation certificate stored as '/home/erfanhosseinabadi/.gnupg/openpgp-revocs.d/F56B518FDB8604BD8BCE01F920754C524A2.rev'
public and secret key created and signed.

pub   rsa4096 2025-03-07 [SC]
      E4FFCF56B518FDB8604BD8BCE01F920754C524A2
uid           erfanhosseinabadi <erfnho3einabadi@gmail.com>
sub   rsa4096 2025-03-07 [E]

[erfanhosseinabadi@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: 1  signed: 0  trust: 0-, 0q, 0n, 0m, 0f, 1u
[keyboard]
-----
sec   rsa4096/E01F920754C524A2 2025-03-07 [SC]
      E4FFCF56B518FDB8604BD8BCE01F920754C524A2
uid           erfanhosseinabadi <erfnho3einabadi@gmail.com>
ssb   rsa4096/5DE89739667CAF51 2025-03-07 [E]

[erfanhosseinabadi@vbox ~]$ gpg --armor --export <PGP Fingerprint>

```

width:70%}

width:70%}

Чтобы скопировать ваш сгенерированный PGP ключ в буфер обмена, используйте:

```
gpg --armor --export ваш_email@example.com
```

```

bash: syntax error near unexpected token `newline'
[erfanhosseinabadi@vbox ~]$ gpg --armor --export E01F920754C524A2
-----BEGIN PGP PUBLIC KEY BLOCK-----

```

{#fig:006

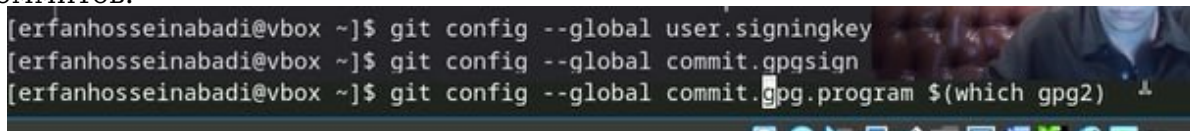
width:70%}

## 4.4 4. Настройка подписей Git

Для настройки автоматической подписи коммитов используйте следующие команды:

```
git config --global commit.gpgSign true
git config --global user.signingkey ваш_PGP_ключ
```

Эти команды позволят **Git** использовать указанный PGP ключ для подписи КОММИТОВ.

A terminal window showing three git config commands being executed. The user is 'erfanhosseinabadi' on a 'vbox' machine. The commands are: 'git config --global user.signingkey', 'git config --global commit.gpgsign', and 'git config --global commit.gpg.program \$(which gpg2)'. The output of the last command is 'gpg2'.

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

{#fig:007

A terminal window showing the 'gh auth login' process. It prompts for where to use GitHub (GitHub.com), preferred protocol (SSH), upload SSH public key (yes), title for SSH key (fedora sway), and how to authenticate (login with a web browser). It then displays a one-time code '2813-666E' and a URL to open in a browser. The terminal also shows some error messages from 'GFX1' related to 'glxtest' and 'ManageChildProcess' failed.

```
[erfanhosseinabadi@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/erfanhosseinabadi/.ssh/id_rsa.pub
? Title for your SSH key: fedora sway
? How would you like to authenticate GitHub CLI? Login with a web browser

! First copy your one-time code: 2813-666E
Press Enter to open https://github.com/login/device in your browser...
[GFX1-]: ManageChildProcess(glxtest): poll failed: Success
Created with Wondershare DemoCreator
[GFX1-]: glxtest: ManageChildProcess failed
[GFX1-]: No GPUs detected via PCI
```

width:70%}

{#fig:008 width:70%}

## 4.5 5. Регистрация на GitHub

Необходимо авторизоваться на **GitHub**. Перейдите на сайт и выполните вход в свой аккаунт.



```

1 gpg.program $(which gpg2)
[erfanhosseinabadi@vbox ~]$ 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/erfanhosseinabadi/.ssh/id_rsa.pub
? Title for your SSH key: fedora sway
? How would you like to authenticate GitHub CLI? Login with a web browser

! First copy your one-time code: 2813-666E
Press Enter to open https://github.com/login/device in your browser...
[GFX1-]: ManageChildProcess(glxtest): poll failed: Success
Created with
[GFX1-]: glxtest: ManageChildProcess failed
Wondershare DemoCreator f
[GFX1-]: No GPUs detected via PCI

```

{#fig:009 width:70%}

## 4.6 6. Создание локального каталога для выполнения заданий

Создайте локальный каталог для хранения ваших заданий по предмету:

`mkdir` название\_каталога

`cd` название\_каталога

Мы создали каталог для нашего курса.

```

[erfanhosseinabadi@vbox ~]$ mkdir -p ~/work/study/2022-2023/"Операционные системы"
[erfanhosseinabadi@vbox ~]$ cd ~/work/study/2022-2023/"Операционные системы"

```

{#fig:010

width:70%}

## 4.7 Использование репозитория

После создания каталога вы можете использовать репозиторий **yamadharma**:

```
erfanhosseinabadi@vbox Операционные системы]$ gh repo create study_2022-2023_os-intro --template=yamadharma/cour
se-directory-student-template --public
Created repository erfanhosseinabadi/study_2022-2023_os-intro on GitHub
https://github.com/erfanhosseinabadi/study_2022-2023_os-intro
erfanhosseinabadi@vbox Операционные системы]$ git clone --recursive git@github.com:<owner>/study_2022-2023_os-in
tro.git os-intro
bash: owner: No such file or directory
erfanhosseinabadi@vbox Операционные системы]$ cd
erfanhosseinabadi@vbox ~]$ ls
Desktop Documents Downloads Music Pictures Public Templates Videos wo
erfanhosseinabadi@vbox ~]$ mkdir -p ~/work/study/2022-2023/"Операционные сис
erfanhosseinabadi@vbox ~]$ cd ~/work/study/2022-2023/"Операционные системы"
erfanhosseinabadi@vbox Операционные системы]$ git clone --recursive study_2022-2023
se-directory-student-template --public
GraphQL: Could not clone: Name already exists on this account (cloneTemplateR
erfanhosseinabadi@vbox Операционные системы]$ mkdir -p ~/work/study/2022-202
```



{#fig:011

width:70%}

### 4.7.1 Удаление лишних файлов

Удалите ненужные файлы с помощью команды:

`rm` ненужный\_файл

```
erfanhosseinabadi@vbox Операционные системы]$ cd os-intro
erfanhosseinabadi@vbox os-intro]$ rm package.json
```

{#fig:012 width:70%}

### 4.7.2 Создание необходимых каталогов

Создайте все необходимые подкаталоги для организации вашего проекта:

`mkdir` подкаталог1 подкаталог2



```

[erfanhosseinabadi@vbox os-intro]$ echo os-intro > COURSE
[erfanhosseinabadi@vbox os-intro]$ make
Usage:
  make <target>

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

[erfanhosseinabadi@vbox os-intro]$ git add .
[erfanhosseinabadi@vbox os-intro]$ git commit -am 'feat(main): make course structure'
[master f97c7be] feat(main): make course structure
2 files changed, 1 insertion(+), 14 deletions(-)
delete mode 100644 package.json
[erfanhosseinabadi@vbox os-intro]$ git push
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 4 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 955 bytes | 955.00 KiB/s, done.
Total 3 (delta 1), reused 0 (delta 0), pushed 3 (delta 1)
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To github.com:erfanhosseinabadi/study_2022-2023_os-intro.git
  4e1e9c0..f97c7be master -> master
[erfanhosseinabadi@vbox os-intro]$

```

{#fig:013}

width:70%}

## 4.8 Выводы

В результате выполнения задания вы:

- Настроили **Git** с вашим именем и email.
- Сгенерировали и добавили SSH ключ на **GitHub**.
- Создали PGP ключ и настроили автоматическую подпись коммитов.
- Зарегистрировались на **GitHub** и создали локальный каталог для работы.

Эти шаги являются основными для успешной работы с **Git** и **GitHub**.

В этой лаборатории с помощью команд мы создаем новый репозиторий. Мы также смогли создать новый каталог с информацией о новом курсе. Как связать нашу учетную запись Github, чтобы мы могли работать, а также как сохранять, а затем загружать документы.

## 5 Ответы на контрольные вопросы.

1. Что такое системы контроля версий (VCS) и для решения каких задач они предназначаются? Системы контроля версий (VCS) — это инструменты, которые позволяют управлять изменениями в коде, документах или других файлах в проекте. Они решают следующие задачи:
2. Отслеживание изменений файлов.
3. Хранение истории изменений.
4. Совместная работа над проектом.
5. Возможность отката к предыдущим версиям.
6. Разрешение конфликтов при одновременном редактировании.
7. Объясните следующие понятия VCS и их отношения: хранилище, commit, история, рабочая копия.
8. Что представляют собой и чем отличаются централизованные и децентрализованные VCS? Приведите примеры VCS каждого вида.
9. Опишите действия с VCS при единоличной работе с хранилищем.
10. Опишите порядок работы с общим хранилищем VCS.
11. Каковы основные задачи, решаемые инструментальным средством git?
12. Назовите и дайте краткую характеристику командам git.
13. Приведите примеры использования при работе с локальным и удалённым репозиториями.
14. Что такое и зачем могут быть нужны ветви (branches)?
15. Как и зачем можно игнорировать некоторые файлы при commit?