

Шаблон отчёта по лабораторной работе №2

Первоначальна настройка git

Абдуллахи Шугофа

Содержание

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

3.0.1 Установка программного обеспечения

- устоновим git

```
[sabdullakhi@sabdullakhi ~]$ sudo dnf install git
[sudo] password for sabdullakhi:
Fedora 39 - x86_64 - Updates      12 kB/s | 19 kB    0%
Fedora 39 - x86_64 - Updates     671 kB/s | 2.9 MB   0%
Last metadata expiration check: 0:00:18 ago on Thu 29 Feb 2024 0
PM MSK.
Package git-2.44.0-1.fc39.x86_64 is already installed.
Dependencies resolved.
Nothing to do.
Complete!
```

Рис. 1: git устоновки

- Установка gh

```
[sabdullakhi@sabdullakhi ~]$ sudo dnf install gh
Last metadata expiration check: 0:00:38 ago on Thu 29 Feb 2024 01:02:27 PM MSK.
Dependencies resolved.
=====
Package      Architecture Version           Repository      Size
=====
Installing:
gh           x86_64          2.43.1-1.fc39    updates        9.1 M
Transaction Summary
=====
Install 1 Package

Total download size: 9.1 M
Installed size: 46 M
Is this ok [y/N]: y
Downloading Packages:
gh-2.43.1-1.fc39.x86_64.rpm                2.1 MB/s | 9.1 MB    00:04
-----
Total                                       1.9 MB/s | 9.1 MB    00:04
Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction
  Preparing      :                                1/1
  Installing     : gh-2.43.1-1.fc39.x86_64        1/1
  Running scriptlet: gh-2.43.1-1.fc39.x86_64        1/1
  Verifying      : gh-2.43.1-1.fc39.x86_64        1/1

Installed:
gh-2.43.1-1.fc39.x86_64

Complete!
```

Рис. 2: gh установка

3.0.2 Базовая настройка git

- зададим имя и email владельца репозитория:
- настроим utf-8 в выводе сообщений git:
- Настройте верификацию и подписание коммитов git (см. Верификация коммитов git с помощью GPG).
- Зададим имя начальной ветки (будем называть её master):
- Параметр autocrlf:
- Параметр safecrlf:

```
[sabdullakhi@sabdullakhi ~]$ git config --global user.name "Shogofa21"
[sabdullakhi@sabdullakhi ~]$ git config --global user.email "abdullahishogofa11@gmail.com"
[sabdullakhi@sabdullakhi ~]$ git config --global core.quotepath false
[sabdullakhi@sabdullakhi ~]$ git config --global init.defaultBranch master
[sabdullakhi@sabdullakhi ~]$ git config --global core.autocrlf input
[sabdullakhi@sabdullakhi ~]$ git config --global core.safecrlf warn
```

Рис. 3: задала имя, email, настроила utf-8, имя master, autocrlf и safecrlf

3.1 Создайте ключи ssh

- по алгоритму rsa с ключём размером 4096 бит:

```
[sabdullakhi@sabdullakhi ~]$ ssh-keygen -t rsa -b 4096
Generating public/private rsa key pair.
Enter file in which to save the key (/home/sabdullakhi/.ssh/id_rsa):
Created directory '/home/sabdullakhi/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/sabdullakhi/.ssh/id_rsa
Your public key has been saved in /home/sabdullakhi/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:n5WfhmzCpciql1iKY+fRhIHIXdZQgxhy4D1V1m2Iqo sabdullakhi@sabdullakhi
The key's randomart image is:
+---[RSA 4096]-----+
|o*=+o. .oo      |
|+= +o. . . .    |
|.o . o . .      |
|. . o . . . .   |
|o . S +         |
|.o.. . + * o .  |
|=E= o * + +     |
|+.o.o . o .     |
|o++ .o.         |
+---[SHA256]-----+
[sabdullakhi@sabdullakhi ~]$
```

Рис. 4: создала ключ в рвзмере 4096 бит

- по алгоритму ed25519:

```
[sabdullakhi@sabdullakhi ~]$ ssh-keygen -t ed25519
Generating public/private ed25519 key pair.
Enter file in which to save the key (/home/sabdullakhi/.ssh/id_ed25519):

Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/sabdullakhi/.ssh/id_ed25519
Your public key has been saved in /home/sabdullakhi/.ssh/id_ed25519.pub
The key fingerprint is:
SHA256:12u+LLh1I4iF76qjbW8U4u5QcUUe8FH13HiszIV0Tqo sabdullakhi@sabdullak
hi
The key's randomart image is:
+--[ED25519 256]--+
|      .o=... . o|
|      + o  + X |
|      . . o    * +|
|      .o.. . . + +|
|      .....S . E +|
|      .. + o   . |
|      .. . o.o =  |
|      o+ .....* . |
|      .+=o.o. .+. |
+-----[SHA256]-----+
[sabdullakhi@sabdullakhi ~]$
```

Рис. 5: создала ключ по алгоритму ed25519

3.2 Создайте ключи gpg

- Генерируем ключ
- Из предложенных опций выбираем:
 - выбрала тип RSA and RSA
 - размер 4096
 - выберила срок действия; значение по умолчанию — 0
- GPG запросит личную информацию, которая сохранится в ключе:
 - написала имя Shogofa Abdullahi
 - написала электронной почты abdullahishogofa11@gmail.com
- При вводе email убедитесь, что он соответствует адресу, используемому на GitHub.
 - Комментарий. Можно ввести что угодно или нажать клавишу ввода, чтобы оставить это поле пустым.

```
[sabdullakhi@sabdullakhi ~]$ gpg --full-generate-key

gpg (GnuPG) 2.4.3; Copyright (C) 2023 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/sabdullakhi/.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

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

Real name: Shogofa Abdullahi
Email address: abdullahishogofa1@gmail.com
Comment:
You selected this USER-ID:
  "Shogofa Abdullahi <abdullahishogofa1@gmail.com>"

Change (N)ame, (C)omment, (E)mail or (O)kay/(Q)uit? o
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.
█
```



Рис. 6: генерируем ключ gpg

3.2.1 Настройка github

у меня уже был github, по этому не была надо опыят настройка github.

3.2.2 Добавление PGP ключа в GitHub

- Выводим список ключей и копируем отпечаток приватного ключа:

```
[sabdullakhi@sabdullakhi ~]$ 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/A9C4FDFC6F14FB96 2024-02-29 [SC]
      E3248433039AB61ABBBFD867A9C4FDFC6F14FB96
uid           [ultimate] Shogofa Abdullahi <abdullahishogofa1@gmail.com>
ssb   rsa4096/7A1DFB2F1E6259C9 2024-02-29 [E]
```

- Выводим список ключей и копируем отпечаток приватного ключа:

```
complete:
[sabdullakhi@sabdullakhi ~]$ gpg --armor --export abduallahishogofa11@gmail.com | xclip -sel clip
[sabdullakhi@sabdullakhi ~]$ gpg --armor --export abduallahishogofa11@gmail.com | cat
-----BEGIN PGP PUBLIC KEY BLOCK-----

mQINBGXgV6UBEADCjPi69Bow4apmzM6y0CszyLcrxfnR1tafCT3JwAX41heq8pos
4kR1zjKQ1/TeGjR9LJntrM/av2jciRUZJgG4CqhnVVE6NWrxXbxR0HW2J7F14EQav
PnUd+950KnEtWxV331d81CuM+35Piq7RR8R+ocx2nZWHPndH1TyCu3cApI3Moeg
/6xB27L1R6534oc6/6EVe4Ruu/DspVDDGT1IHxquX3r4NgeZdCzb5hyhy1wZ0AYV
7grLfARm0o+BRrcE4qxx+FMfzbd2Fa0jmTrys0hgF1+TT33Ugd1jCKt9P+zfGuhe
votIabYfk75vJDeXUGELemEXk7nB2Arw0f7I6A2I1WZ0/ynBsvT9NUxqT8cAw2B6
BRL+5WK1gn4CRVf6to602kBzA/rPTeB+uSJMEFGg+HqYsM7QLdYGkdHe+zqvxFE
0BKR1xPa0upB9xLS0vyq2h1LyMT1Z0zogmq7h+3CJxiDs1YX83yU28JfQeLmPtgV
b0tYs8SDXyAsjf8Z2003+mC3sDIGgFL0Ho3eeeJFTYgXCcX2yzzZuFo8AYxFzfST
PYGmRtHd4HkdXgUwJY0ujWAKvsgS110knAsd3Zvb1VVntnEXIoTxnD9PCuWmtwFt
```

Рис. 7: скопировала gpg ключ

- Перешла в настройки GitHub (<https://github.com/settings/keys>), нажала на кнопку New GPG key и вставляла полученный ключ в поле ввода.

Add new GPG key

Title

OS

Key

```
fknRiypI0YEpmZIMqNek6MZBmBgMFGvQ1RLNQm//
LV2vLemgsiaYR0+AuIzVV42qq
giDeKue9QiNSZ+xTokwB7MCzCbtv/
CnVQaLOOdI1sEQpwvsl+3KTfbvsVIIcLyXc
mq7x6D6ShoTB5oV+nSqHNjtDzN9cxothWO7B46kZIKxRppAhm1hk
m8iMgbr5LwQZ
V2tuOMuoemDXhdomcPb
=ToY7
-----END PGP PUBLIC KEY BLOCK-----
```

Add GPG key

Terms Privacy Security Status Docs Contact Manage cookies

Рис. 8: создала gpg ключ

3.3 Настройка автоматических подписей коммитов git

- Используя введённый email, укажите Git применять его при подписи КОММИТОВ:

```
[sabdullakhi@sabdullakhi ~]$ git config --global user.signingkey abdulla  
hishogofa11@gmail.com  
[sabdullakhi@sabdullakhi ~]$ git config --global commit.gpgsign true  
[sabdullakhi@sabdullakhi ~]$ git config --global gpg.program $(which gpg  
2)
```

Рис. 9: написала email и остольные команды

3.3.1 Настройка gh

- Для начала необходимо авторизоваться

```
sabdullakhi@sabdullakhi ~]$ 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/sabdullakhi/.  
ssh/id_rsa.pub  
? Title for your SSH key: os  
? How would you like to authenticate GitHub CLI? Login with a web browse  
?  
First copy your one-time code: CF2F-BC98  
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/sabdullakhi/.ssh/id  
rsa.pub  
/ Logged in as Shogofa21
```

Рис. 10: настройка gh, и ответа на несколько вопросов

- Авторизоваться можно через броузер.

4 Сознание репозитория курса на основе шаблона

- Создаём репозиторий курса на основе шаблона. Все нужные команды для создания были в указаниях к лабораторной работе. В 4 команде, вместо , указываем своё имя профиля на github.

```
[sabdullakhi@sabdullakhi ~]$ mkdir -p ~/work/study/2022-2023/"Операционные сист
емы"
[sabdullakhi@sabdullakhi Операционные системы]$ gh repo create study_2022-2023_
os-intro --template=yamadharma/course-directory-student-template --public
/ Created repository Shogofa21/study_2022-2023_os-intro on GitHub
[sabdullakhi@sabdullakhi Операционные системы]$ git clone --recursive git@github
b.com:<owner>/study_2022-2023_os-intro.git os-intro
bash: owner: No such file or directory
[sabdullakhi@sabdullakhi Операционные системы]$ git clone --recursive git@github
b.com:Shogofa21/study_2022-2023_os-intro.git os-intro
Cloning into 'os-intro'...
The authenticity of host 'github.com (140.82.121.4)' can't be established.
ED25519 key fingerprint is SHA256:+DiY3wvV6TuJJhbpZisF/zLDA0zPMSvHdkr4UvC0qU.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
```

Рис. 11: сздаала репозитории

4.1 Настройка каталога курса

- Настраиваем каталог курса. Для этого переходим в него командой: `cd ~/work/study/2021-2022/"Операционные системы"/os-intro` Далее командой `ls` проверяем, что мы в него перешли. В каталоге "os-intro" нам потребуется удалить файл "package.json". Выполняем данную задачу командой: `rm package.json`

```
[sabdullakhi@sabdullakhi Операционные системы]$ cd ~/work/study/2022-20
23/"Операционные системы"/os-intro
[sabdullakhi@sabdullakhi os-intro]$ rm package.json
```

Рис. 12: настроила каталога курса

- Создаём необходимые каталоги и отправляем наши файлы на сервер.

```
[sabdullakhi@sabdullakhi os-intro]$ echo os-intro > COURSE
[sabdullakhi@sabdullakhi os-intro]$ make
Usage:
  make <target>

Targets:
  list                List of courses
  prepare             Generate directories structure
  submodule           Update submules
[sabdullakhi@sabdullakhi os-intro]$ make prepare
```

Рис. 13: Создайте необходимые каталоги

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

Рис. 14: Отправьте файлы на сервер

```
create mode 100755 project-personal/stage6/report/pandoc/filters/pa
c_tablenos.py
create mode 100644 project-personal/stage6/report/pandoc/filters/pa
cxnos/__init__.py
create mode 100644 project-personal/stage6/report/pandoc/filters/pa
cxnos/core.py
create mode 100644 project-personal/stage6/report/pandoc/filters/pa
cxnos/main.py
create mode 100644 project-personal/stage6/report/pandoc/filters/pa
cxnos/pandocattributes.py
create mode 100644 project-personal/stage6/report/report.md
[sabdullakhi@sabdullakhi os-intro]$ git push
Enumerating objects: 40, done.
Counting objects: 100% (40/40), done.
Delta compression using up to 4 threads
Compressing objects: 100% (30/30), done.
Writing objects: 100% (17/17), done.
```

Рис. 15: Отправьте файлы на сервер

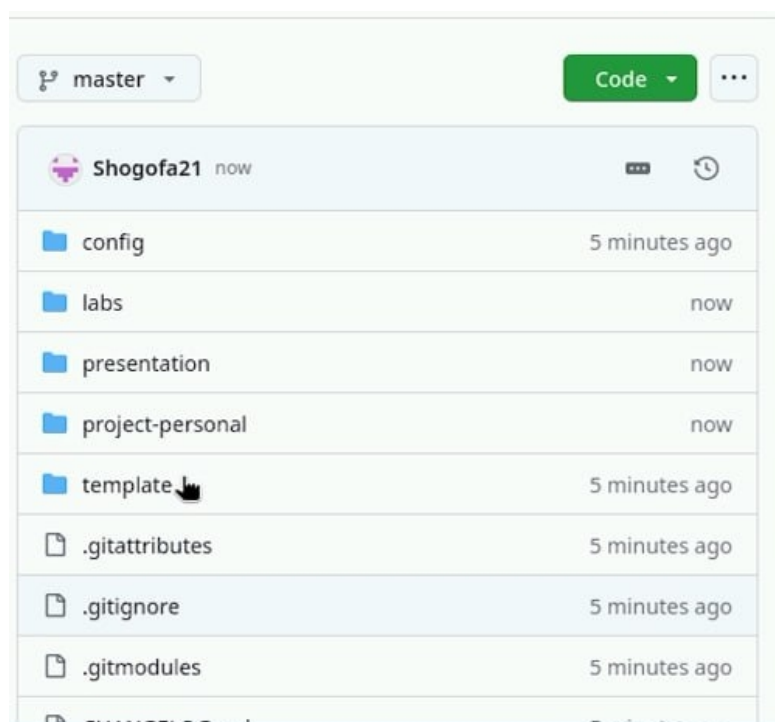


Рис. 16: все готов и правильно

5 Контрольные вопросы:

1. Что такое системы контроля версий (VCS) и для решения каких задач они предназначены?
 - Это программное обеспечение для облегчения работы с изменяющейся информацией. VCS позволяет хранить несколько версий одного и того же документа, при необходимости возвращаться к более ранним версиям, определять, кто и когда сделал то или иное изменение, и многое другое.
2. Объясните следующие понятия VCS и их отношения: хранилище, commit, история, рабочая копия
 - Хранилище (repository), или репозиторий, — место хранения всех версий и служебной информации. Commit («[трудовой] вклад», не переводится) — синоним версии; процесс создания новой версии. История – место, где сохраняются все коммиты, по которым можно посмотреть данные о коммитах. Рабочая копия – текущее состояние файлов проекта, основанное на версии, загруженной из хранилища.
3. Что представляют собой и чем отличаются централизованные и децентрализованные VCS?
 - Приведите примеры VCS каждого вида.
 - Централизованные VCS: одно основное хранилище всего проекта и каждый пользователь копирует себе необходимые ему файлы из этого репозитория, изменяет и, затем, добавляет свои изменения обратно. Децентрализованные VCS: у каждого пользователя свой вариант (возможно не один) репозитория.
4. Опишите действия с VCS при единоличной работе с хранилищем.
5. Опишите порядок работы с общим хранилищем VCS.
6. Каковы основные задачи, решаемые инструментальным средством git?
 - Git — это система управления версиями. У Git две основных задачи: первая — хранить информацию о всех изменениях в вашем коде, начиная с самой первой строчки, а вторая — обеспечение удобства командной работы над кодом.
7. Назовите и дайте краткую характеристику командам git.
 - `git -version` (Проверка версии Git)
 - `git init` (Инициализировать ваш текущий рабочий каталог как Git-репозиторий)
 - `git clone https://www.github.com/username/repo-name` (Скопировать существующий удаленный Git-репозиторий)
 - `git remote` (Просмотреть список текущих удалённых репозиториях Git)
 - `git remote -v` (Для более подробного вывода)
 - `git add my_script.py` (Можете указать в команде конкретный файл).

- `git add .` (Позволяет охватить все файлы в текущем каталоге, включая файлы, чье имя начинается с точки)
 - `git commit -am "Commit message"` (Вы можете сжать все индексированные файлы и отправить коммит).
 - `git branch` (Просмотреть список текущих веток можно с помощью команды `branch`)
 - `git -help` (Чтобы узнать больше обо всех доступных параметрах и командах)
 - `git push origin master` (Передать локальные коммиты в ветку удаленного репозитория).
8. Приведите примеры использования при работе с локальным и удалённым репозиториями.
 9. Что такое и зачем могут быть нужны ветви (branches)?
 - 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. Как и зачем можно игнорировать некоторые файлы при `commit`?
 - Игнорируемые файлы — это, как правило, артефакты сборки и файлы, генерируемые машиной из исходных файлов в вашем репозитории, либо файлы, которые по какой-либо иной причине не должны попадать в коммиты.

6 Выводы

В ходе выполнения лабораторной работы изучили идеологию и применение средств контроля версий, а также освоили умения по работе с `git`.