

# **Отчёт по лабораторной работе №1**

**Дисциплина: Научное программирование**

Полиенко Анастасия Николаевна, НПМмд-02-23

# Содержание

<b>1</b>	<b>Цель работы</b>	<b>4</b>
<b>2</b>	<b>Задание</b>	<b>5</b>
<b>3</b>	<b>Теоретическое введение</b>	<b>6</b>
<b>4</b>	<b>Выполнение лабораторной работы</b>	<b>8</b>
<b>5</b>	<b>Выводы</b>	<b>12</b>
	<b>Список литературы</b>	<b>13</b>

## Список иллюстраций

4.1	Установка git-flow . . . . .	8
4.2	Установка gh . . . . .	8
4.3	Базовая настройка git . . . . .	8
4.4	Ключ SSH по алгоритму rsa . . . . .	9
4.5	Ключ SSH по алгоритму ed25519 . . . . .	9
4.6	Ключ PGP . . . . .	10
4.7	Автоматические подписи git . . . . .	10
4.8	Настройка gh . . . . .	10
4.9	Создание репозитория . . . . .	11

# 1 Цель работы

1. Изучить идеологию и применение средств контроля версий.
2. Освоить умения по работе с git.
3. Изучить язык разметки Markdown.

## 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.
7. Создать отчёт по лабораторной работе, используя язык разметки Markdown.

### 3 Теоретическое введение

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 дельтакомпрессию — 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 зависимости 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.

Более подробно см. в [1–6].

## 4 Выполнение лабораторной работы

1. Устанавливаем git-flow (рис. 4.1) и gh (рис. 4.2).

```
[anpolienko@anpolienko tmp]$ wget --no-check-certificate -q https://raw.githubusercontent.com/petervanderdoes/gitflow/develop/contrib/gitflow-installer.sh
[anpolienko@anpolienko tmp]$ chmod +x gitflow-installer.sh
[anpolienko@anpolienko tmp]$ sudo ./gitflow-installer.sh install stable
[sudo] password for anpolienko:
### git-flow no-make installer ###
Installing git-flow to /usr/local/bin
```

Рис. 4.1: Установка git-flow

```
[anpolienko@anpolienko tmp]$ sudo dnf install gh
packages for the GitHub CLI                               4.7 kB/s | 2.6 kB    00:00
Dependencies resolved.
=====
Package      Architecture Version      Repository      Size
=====
Installing:
gh            x86_64      2.34.0-1     gh-cli          11 M
Transaction Summary
=====
```

Рис. 4.2: Установка gh

1. Создаём базовую конфигурацию для работы с git (рис. 4.3).

```
[anpolienko@anpolienko tmp]$ git config --global user.name "Anastasiya Polienko"
[anpolienko@anpolienko tmp]$ git config --global user.email "1132236935@rudn.ru"
[anpolienko@anpolienko tmp]$ git config --global core.quotepath false
[anpolienko@anpolienko tmp]$ git config --global init.defaultBranch master
[anpolienko@anpolienko tmp]$ git config --global core.autocrlf input
[anpolienko@anpolienko tmp]$ git config --global core.safecrlf warn
```

Рис. 4.3: Базовая настройка git

1. Создаём ключи SSH по алгоритму rsa (рис. 4.4) и алгоритму ed25519 (рис. 4.5) и ключ PGP (рис. 4.6).



```
[anpolienko@anpolienko tmp]$ ssh-keygen -t rsa -b 4096
Generating public/private rsa key pair.
Enter file in which to save the key (/home/anpolienko/.ssh/id_rsa):
Created directory '/home/anpolienko/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/anpolienko/.ssh/id_rsa.
Your public key has been saved in /home/anpolienko/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:FRM/0Ya7cIoxgLErR4lIY06P68Rxc4qglpMhQE6sTTU anpolienko@anpolienko.localdo
main
The key's randomart image is:
+---[RSA 4096]-----+
|+..E      +.      |
|Bo.*      =.      |
|+== o      o *     |
|+...      . o o    |
|++oo .o S o        |
|=oX + + + .        |
|.X o      . .      |
|o o              |
|.o              |
+-----[SHA256]-----+
```

Рис. 4.4: Ключ SSH по алгоритму rsa

```
[anpolienko@anpolienko tmp]$ ssh-keygen -t ed25519
Generating public/private ed25519 key pair.
Enter file in which to save the key (/home/anpolienko/.ssh/id_ed25519):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/anpolienko/.ssh/id_ed25519.
Your public key has been saved in /home/anpolienko/.ssh/id_ed25519.pub.
The key fingerprint is:
SHA256:RM1m1W9VPYtMk77f932R0Goeq26Z0VjaQb9g7p5YrSk anpolienko@anpolienko.localdo
main
The key's randomart image is:
+--[ED25519 256]--+
|      .o ... +|
|      . = +..o|
|      .o + +.+|
|      . . = oo|
|      S . . +..|
|      =.* o |
|      B.0.= o|
|      Eo0++ .|=|
|      .*0o  *|
+-----[SHA256]-----+
```

Рис. 4.5: Ключ SSH по алгоритму ed25519

```
[anpolienko@anpolienko tmp]$ gpg --full-generate-key
gpg (GnuPG) 2.2.20; Copyright (C) 2020 Free Software Foundation, Inc.
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/anpolienko/.gnupg' created
gpg: keybox '/home/anpolienko/.gnupg/pubring.kbx' created
Please select what kind of key you want:
  (1) RSA and RSA (default)
  (2) DSA and Elgamal
  (3) DSA (sign only)
  (4) RSA (sign only)
  (14) Existing key from card
Your selection? 1
RSA keys may be between 1024 and 4096 bits long.
What keysizes do you want? (2048) 4096
Requested keysizes 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.
```

Рис. 4.6: Ключ PGP

1. Настраиваем автоматические подписи коммитов git (рис. 4.7).

```
[anpolienko@anpolienko Научное программирование]$ git config --global user.sign
ingkey BBB38E80AC836535
[anpolienko@anpolienko Научное программирование]$ git config --global commit.gp
gsign true
[anpolienko@anpolienko Научное программирование]$ git config --global gpg.progr
am $(which gpg2)
```

Рис. 4.7: Автоматические подписи git

1. Авторизуемся с системе gh (рис. 4.8).

```
[anpolienko@anpolienko tmp]$ gh auth login
? What account do you want to log into? GitHub.com
? What is your preferred protocol for Git operations? HTTPS
? Authenticate Git with your GitHub credentials? Yes
? How would you like to authenticate GitHub CLI? Login with a web browser
```

Рис. 4.8: Настройка gh

1. Создадим репозиторий курса на основе шаблона для работы (рис. 4.9).

```

[anpolienko@anpolienko tmp]$ mkdir -p ~/work/study/2023-2024/"Научное программирование"
[anpolienko@anpolienko tmp]$ свввв ~/work/study/2023-2024/"Научное программирование"
bash: свввв: command not found...
^[[A^[[A^C
[anpolienko@anpolienko tmp]$ cd ~/work/study/2023-2024/"Научное программирование"
[anpolienko@anpolienko Научное программирование]$ gh repo create study_2023-2024_sciprog --template=yamadharma/course-directory-student-template --public
✓ Created repository anpolienko/study_2023-2024_sciprog on GitHub
[anpolienko@anpolienko Научное программирование]$ git clone --recursive git@github.com:anpolienko/study_2023-2024_sciprog.git sciprog

```

Рис. 4.9: Создание репозитория

## 5 Выводы

Изучила идеологию и применение средств контроля версий, освоила умения по работе с git, изучила язык разметки Markdown.

## Список литературы

1. GNU Bash Manual [Электронный ресурс]. Free Software Foundation, 2016.  
URL: <https://www.gnu.org/software/bash/manual/>.
2. Newham C. Learning the bash Shell: Unix Shell Programming. O'Reilly Media, 2005. 354 с.
3. Zarrelli G. Mastering Bash. Packt Publishing, 2017. 502 с.
4. Robbins A. Bash Pocket Reference. O'Reilly Media, 2016. 156 с.
5. Таненбаум Э. Архитектура компьютера. 6-е изд. СПб.: Питер, 2013. 874 с.
6. Таненбаум Э., Бос Х. Современные операционные системы. 4-е изд. СПб.: Питер, 2015. 1120 с.