

# **Отчёт по лабораторной №2**

**Дисциплина: Операционные системы**

Воронов Александр Валерьевич

# Содержание

|          |                                       |           |
|----------|---------------------------------------|-----------|
| <b>1</b> | <b>Цель работы</b>                    | <b>5</b>  |
| <b>2</b> | <b>Теоретическое введение</b>         | <b>6</b>  |
| <b>3</b> | <b>Выполнение лабораторной работы</b> | <b>7</b>  |
| <b>4</b> | <b>Контрольные вопросы</b>            | <b>10</b> |
|          | <b>Список литературы</b>              | <b>14</b> |

# Список иллюстраций

|      |  |   |
|------|--|---|
| 3.1  | Базовая настройка git . . . . .                          | 7 |
| 3.2  | Добавление SSH ключа на github . . . . .                 | 7 |
| 3.3  | Генерация GPG ключа . . . . .                            | 8 |
| 3.4  | Копирование GPG ключ . . . . .                           | 8 |
| 3.5  | Добавление GPG ключа на github . . . . .                 | 8 |
| 3.6  | Настройка автоматических подписей коммитов git . . . . . | 8 |
| 3.7  | Настройка gh . . . . .                                   | 9 |
| 3.8  | Создание репозитория . . . . .                           | 9 |
| 3.9  | Создание файлов для отправки . . . . .                   | 9 |
| 3.10 | Отправление файлов на сервер . . . . .                   | 9 |

## **Список таблиц**

# 1 Цель работы

Изучить идеологию и применение средств контроля версий Освоить умения по работе с git # 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.

## 2 Теоретическое введение

Системы контроля версий (Version Control System, VCS) применяются при работе нескольких человек над одним проектом. Обычно основное дерево проекта хранится в классических системах контроля версий используется централизованная модель, предполагающая наличие единого репозитория для хранения файлов. Выполнен

Системы контроля версий поддерживают возможность отслеживания и разрешения конфликтов, которые могут возникнуть при работе нескольких человек над одним

Системы контроля версий также могут обеспечивать дополнительные, более гибкие функциональные возможности. Например, они могут поддерживать работу с не

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

### 3 Выполнение лабораторной работы

Задаем имя и email владельца репозитория, настраиваем utf-8 в выводе сообщений(рис.3.1)

```
git config --global user.name "Alexander Voronov"  
git config --global user.email "voron7741@gmail.com"  
git config --global core.quotePath false  
sudo dnf install gnupg
```

Рис. 3.1: Базовая настройка git

Создание и добавление SSH ключа на github(рис.3.2)

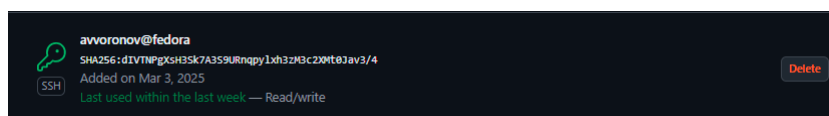


Рис. 3.2: Добавление SSH ключа на github

Генерируем GPG ключ(рис.3.3)

```
root@fedora:~# 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: создан каталог '/root/.gnupg'
Выберите тип ключа:
  (1) RSA and RSA
  (2) DSA and Elgamal
  (3) DSA (sign only)
  (4) RSA (sign only)
  (9) ECC (sign and encrypt) *default*
 (10) ECC (только для подписи)
 (14) Existing key from card
Ваш выбор? 1
длина ключей RSA может быть от 1024 до 4096.
Какой размер ключа Вам необходим? (3072) 4096
Запрошенный размер ключа - 4096 бит
Выберите срок действия ключа.
  0 = не ограничен
  <n> = срок действия ключа - n дней
  <n>w = срок действия ключа - n недель
  <n>m = срок действия ключа - n месяцев
  <n>y = срок действия ключа - n лет
Срок действия ключа? (0) 0
Срок действия ключа не ограничен
Все верно? (y/N) y

GnuPG должен составить идентификатор пользователя для идентификации ключа.
Ваше полное имя: 
```

Рис. 3.3: Генерация GPG ключа

Копируем GPG в буфер обмена(рис.3.4)

```
root@fedora:~# gpg --armor --export C66A832E55912049
[0] 0:sudo*
```

Рис. 3.4: Копирование GPG ключ

Добавляем GPG ключ на github(рис.3.5)



Рис. 3.5: Добавление GPG ключа на github

Используем введенный email, указывая git где будут применять его при подписи коммитов(рис. 3.6)

```
root@fedora:~# git config --global user.signingkey F533CD10250CF6C2A98FB138C66A832E55912049
root@fedora:~# git config --global commit.gpgsign true
root@fedora:~# git config --global gpg.program $(which gpg2)
```

Рис. 3.6: Настройка автоматических подписей коммитов git

Авторизуемся с помощью gh auth login(рис.3.7)



```
avvoronov@fedora:~/work/study/2024-2025/Операционные системы$ 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? Skip
? How would you like to authenticate GitHub CLI? Paste an authentication token
Tip: you can generate a Personal Access Token here https://github.com/settings/tokens
The minimum required scopes are 'repo', 'read:org'.
? Paste your authentication token: *****
- gh config set -h github.com git_protocol ssh
✓ Configured git protocol
✓ Logged in as avvoronov549
avvoronov@fedora:~/work/study/2024-2025/Операционные системы$ gh repo create study_2024-2025_os-intro --template
-yamadharma/course-directory-student-template --public
✓ Created repository avvoronov549/study_2024-2025_os-intro on GitHub
https://github.com/avvoronov549/study_2024-2025_os-intro
```

Рис. 3.7: Настройка gh

Создаем репозиторий курса на основе шаблона(рис.3.8)

```
avvoronov@fedora:~/work/study/2024-2025/Операционные системы$ git clone --recursive git@github.com:avvoronov549/s
tudy_2024-2025_os-intro
Клонирование в «study_2024-2025_os-intro»...
The authenticity of host 'github.com (140.82.121.3)' can't be established.
ED25519 key fingerprint is SHA256:+DiY3wvV6TuJhbpZisF/zLDA0zPKSvHdkr4Uvc0QU.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'github.com' (ED25519) to the list of known hosts.
Enter passphrase for key '/home/avvoronov/.ssh/id_ed25519':
remote: Enumerating objects: 36, done.
remote: Counting objects: 100% (36/36), done.
remote: Compressing objects: 100% (35/35), done.
remote: Total 36 (delta 1), reused 21 (delta 0), pack-reused 0 (from 0)
Получение объектов: 100% (36/36), 19.37 Киб | 944.00 Киб/с, готово.
Определение изменений: 100% (1/1), готово.
Подмодуль «template/presentation» (https://github.com/yamadharma/academic-presentation-markdown-template.git) зар
егистрирован по пути «template/presentation»
Подмодуль «template/report» (https://github.com/yamadharma/academic-laboratory-report-template.git) зарегистриров
ан по пути «template/report»
Клонирование в «/home/avvoronov/work/study/2024-2025/Операционные системы/study_2024-2025_os-intro/template/prese
ntation»...
```

Рис. 3.8: Создание репозитория

Создаем необходимые каталоги(рис.3.9)

```
avvoronov@fedora:~/work/study/2024-2025/Операционные системы/os-intro$ echo os-intro > COURSE
avvoronov@fedora:~/work/study/2024-2025/Операционные системы/os-intro$ make prepare
avvoronov@fedora:~/work/study/2024-2025/Операционные системы/os-intro$ git add .
avvoronov@fedora:~/work/study/2024-2025/Операционные системы/os-intro$ git commit -am 'feat(main): main course st
ructure'
```

Рис. 3.9: Создание файлов для отправки

Отправляем файлы на сервер(рис.3.10)

```
avvoronov@fedora:~/work/study/2024-2025/Операционные системы/os-intro$ git push
Enter passphrase for key '/home/avvoronov/.ssh/id_ed25519':
Перечисление объектов: 40, готово.
Подсчет объектов: 100% (40/40), готово.
При сжатии изменений используется до 2 потоков
Сжатие объектов: 100% (30/30), готово.
Запись объектов: 100% (38/38), 341.66 Киб | 1.95 Миб/с, готово.
Total 38 (delta 4), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (4/4), completed with 1 local object.
To github.com:avvoronov549/study_2024-2025_os-intro
 a00341b..6f3644f master -> master
```

Рис. 3.10: Отправление файлов на сервер

## 4 Контрольные вопросы

Что такое системы контроля версий (VCS) и для решения каких задач они предназначены?

Система контроля версий — программное обеспечение для облегчения работы с изменяющейся информацией. Система управления версиями 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. Системы контроля версий (Version Control System, VCS) применяются для:

- Хранение полной истории изменений
- Причин всех производимых изменений
- Откат изменений, если что-то пошло не так
- Поиск причины и ответственного за появления ошибок в программе
- Совместная работа группы над одним проектом
- Возможность изменять код, не мешая работе других пользователей

2. Объясните следующие понятия VCS и их отношения: хранилище, commit, история, рабочая копия

Репозиторий - хранилище версий - в нем хранятся все документы вместе с историей их изменения и другой служебной информацией. Commit — отслеживание изменений Рабочая копия - копия проекта, связанная с репозиторием (текущее состояние файлов проекта, основанное на версии из хранилища (обычно на последней)) История хранит все изменения в проекте и позволяет при необходимости обратиться к нужным данным.

3. Что представляют собой и чем отличаются централизованные и децентрализованные VCS? Приведите примеры VCS каждого вида

Централизованные VCS (Subversion; CVS; TFS; VAULT; AccuRev): Одно основное хранилище всего проекта Каждый пользователь копирует себе необходимые ему файлы из этого репозитория, изменяет и, затем, добавляет свои изменения обратно

Децентрализованные VCS (Git; Mercurial; Bazaar): У каждого пользователя свой вариант (возможно не один) репозитория Присутствует возможность добавлять и забирать изменения из любого репозитория . В классических системах контроля версий используется централизованная модель, предполагающая наличие единого репозитория для хранения файлов. Выполнение большинства функций по управлению версиями осуществляется специальным сервером. В отличие от классических, в распределённых системах контроля версий центральный репозиторий не является обязательным.

4. Опишите действия с VCS при единоличной работе с хранилищем.

Сначала создаем и подключаем удаленный репозиторий. Затем по мере изменения проекта отправлять эти изменения на сервер.

5. Опишите порядок работы с общим хранилищем VCS.

Участник проекта (пользователь) перед началом работы посредством определенных 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.

6. Каковы основные задачи, решаемые инструментальным средством git?

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.

Наиболее часто используемые команды 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 add имена_файлов` • удалить файл и/или каталог из индекса репозитория (при этом файл и/или каталог остаётся в локальной директории): `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 commit` • создание новой ветки, базирующейся на текущей: `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.

`git push --all (push origin master/любой branch)`

9. Что такое и зачем могут быть нужны ветви (branches)?

Ветвление («ветка», branch) — один из параллельных участков истории в одном хранилище, исходящих из одной версии (точки ветвления). [3]Обычно есть

главная ветка (master), или ствол (trunk). Между ветками, то есть их концами, возможно слияние. Используются для разработки новых функций.

#### 10. Как и зачем можно игнорировать некоторые файлы при commit?

Во время работы над проектом так или иначе могут создаваться файлы, которые не требуется добавлять в последствии в репозиторий. Например, временные файлы, создаваемые редакторами, или объектные файлы, создаваемые компиляторами. Можно прописать шаблоны игнорируемых при добавлении в репозиторий типов файлов в файл .gitignore с помощью сервисов. # Выводы

В результате выполнения лабораторной работы я приобрел навыки работы с гит, научился созданию репозитория, генерации и ssh ключей, настроил каталог курса и авторизовался в gh.

## **Список литературы**