

# **Отчёт по лабораторной работе №2**

**дисциплина: Операционные системы**

Сычев Егор Олегович

# Содержание

<b>1</b>	<b>Цель работы</b>	<b>5</b>
<b>2</b>	<b>Выполнение лабораторной работы</b>	<b>6</b>
<b>3</b>	<b>Контрольные вопросы</b>	<b>11</b>
<b>4</b>	<b>Вывод</b>	<b>15</b>

## **Список иллюстраций**

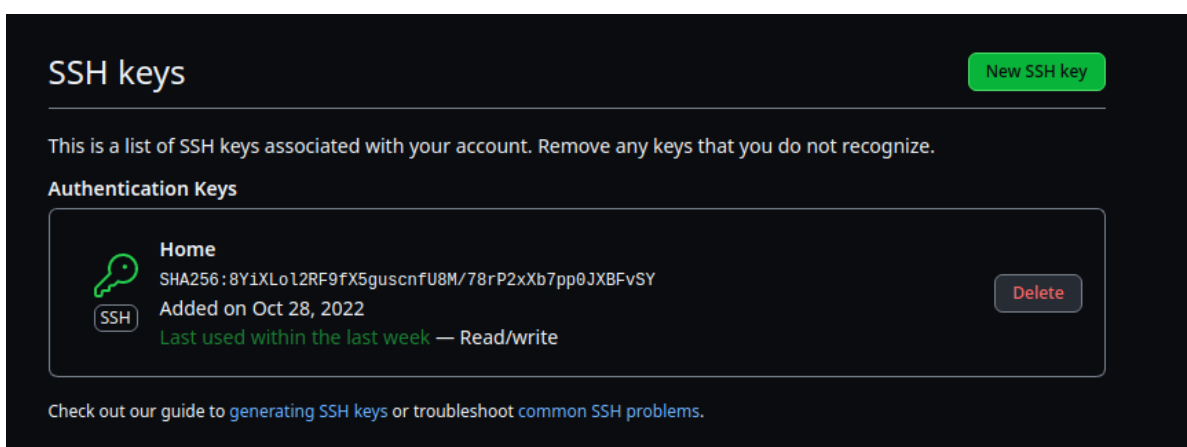
## Список таблиц

# 1 Цель работы

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

## 2 Выполнение лабораторной работы

1. Установка git,gh. Базовая настройка git. Создание ssh ключа. (Выполнено в предыдущих лабораторных работах)



2. Создание pgr ключа.

```
[eosihchev@fedora report]$ gpg --full-generate-key
gpg (GnuPG) 2.3.7; Copyright (C) 2021 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.

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
```

3. Добавление pgr ключа в github.


```
[eosihchev@fedora ~]$ gpg --list-secret-keys --keyid-format LONG
gpg: проверка таблицы доверия
gpg: marginals needed: 3 completes needed: 1 trust model: pgp
gpg: глубина: 0 достоверных: 1 подписанных: 0 доверие: 0-, 0q, 0n, 0m, 0f,
lu
/home/eosihchev/.gnupg/pubring.kbx
-----
sec   rsa4096/3E70D02667284130 2023-02-17 [SC]
      B64DCA7FD64B5B24666252AE3E70D02667284130
uid    [ абсолютно ] Egor <1132226469@pfur.ru>
ssb    rsa4096/1026378F92854477 2023-02-17 [E]

[eosihchev@fedora ~]$ 
[eosihchev@fedora ~]$ gpg --armor --export 3E70D02667284130 | xclip -sel clip
[eosihchev@fedora ~]$
```

## GPG keys

New GPG key

This is a list of GPG keys associated with your account. Remove any keys that you do not recognize.



home

Email address: 1132226469@pfur.ru

Key ID: 3E70D02667284130

Subkeys: 1026378F92854477

Added on Feb 17, 2023

Delete

[Learn how to generate a GPG key and add it to your account.](#)

#### 4. Настройка автоматических подписей коммитов git.

```
[eosihchev@fedora ~]$ git config --global user.signingkey 3E70D02667284130
[eosihchev@fedora ~]$ git config --global commit.gpgsign true
[eosihchev@fedora ~]$ git config --global gpg.program $(which gpg2)
[eosihchev@fedora ~]$
```

#### 5. Настройка gh.

```
[eosihchev@fedora ~]$ gh auth login
bash: gh: команда не найдена...
Установить пакет «gh», предоставляющий команду «gh»? [N/y] y

* Ожидание в очереди...
* Загрузка списка пакетов....
Следующие пакеты должны быть установлены:
gh-2.22.1-1.fc36.x86_64      GitHub's official command line tool
Продолжить с этими изменениями? [N/y] y

* Ожидание в очереди...
* Ожидание аутентификации...
* Ожидание в очереди...
* Загрузка пакетов...
* Запрос данных...
* Проверка изменений...
* Установка пакетов... █
```



**Congratulations, you're all set!**

Your device is now connected.

6. Создание репозитория курса на основе шаблона и настройка каталога курса.



## Create a new repository from course-directory-student-template

The new repository will start with the same files and folders as `yamadharma/course-directory-student-template`

Owner \*



eosihchev

Repository name \*

study\_2022-2023\_os-intro



Great repository names are short and memorable. Need inspiration? How about `fluffy-umbrella?`

Description (optional)



Public

Anyone on the internet can see this repository. You choose who can commit.



Private

You choose who can see and commit to this repository.



Include all branches

Copy all branches from `yamadharma/course-directory-student-template` and not just `master`.

You are creating a public repository in your personal account.

Create repository from template

```
eosihchev@fedora:~/work/study/2022-2023/Операционные системы
[eosihchev@fedora ~]$ mkdir -p ~/work/study/2022-2023/"Операционные системы"
[eosihchev@fedora ~]$ cd ~/work/study/2022-2023/"Операционные системы"
[eosihchev@fedora Операционные системы]$ gh repo create study_2022-2023_os-intro --template=yamadharma/course-directory-student-template --public
GraphQL: Could not clone: Name already exists on this account (cloneTemplateRepository)
[eosihchev@fedora Операционные системы]$ git clone --recursive git@github.com:eosihchev/study_2022-2023_os-intro.git os-intro
Клонирование в «os-intro»...
remote: Enumerating objects: 27, done.
remote: Counting objects: 100% (27/27), done.
remote: Compressing objects: 100% (26/26), done.
remote: Total 27 (delta 1), reused 11 (delta 0), pack-reused 0
Получение объектов: 100% (27/27), 16.93 КиБ | 16.93 КиБ/с, готово.
Определение изменений: 100% (1/1), готово.
Подмодуль «template/presentation» (https://github.com/yamadharma/academic-presentation-markdown-template.git) зарегистрирован по пути «template/presentation»
Подмодуль «template/report» (https://github.com/yamadharma/academic-laboratory-report-template.git) зарегистрирован по пути «template/report»
Клонирование в «/home/eosihchev/work/study/2022-2023/Операционные системы/os-intro/template/presentation»...
remote: Enumerating objects: 82, done.
remote: Counting objects: 100% (82/82), done.
remote: Compressing objects: 100% (57/57), done.
remote: Total 82 (delta 28), reused 77 (delta 23), pack-reused 0
Получение объектов: 100% (82/82), 92.90 КиБ | 980.00 КиБ/с, готово.
Определение изменений: 100% (28/28), готово.
Клонирование в «/home/eosihchev/work/study/2022-2023/Операционные системы/os-intro/template/report»...
remote: Enumerating objects: 101, done.
remote: Counting objects: 100% (101/101), done.
remote: Compressing objects: 100% (70/70), done.
remote: Total 101 (delta 40), reused 88 (delta 27), pack-reused 0
Получение объектов: 100% (101/101), 327.25 КиБ | 1.62 МиБ/с, готово.
Определение изменений: 100% (40/40), готово.
Submodule path 'template/presentation': checked out 'b1be3800ee91f5809264cb755d316174540b753e'
Submodule path 'template/report': checked out '1d1b61dcac9c287a83917b82e3aef11a33b1e3b2'
[eosihchev@fedora Операционные системы]$
```

```
[eosihchev@fedora Операционные системы]$ cd ~/work/study/2022-2023/"Операционные системы"/os-intro
[eosihchev@fedora os-intro]$ rm package.json
[eosihchev@fedora os-intro]$ echo os-intro > COURSE
[eosihchev@fedora os-intro]$ make

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

[eosihchev@fedora os-intro]$ git push
Перечисление объектов: 40, готово.
Подсчет объектов: 100% (40/40), готово.
При сжатии изменений используется до 4 потоков
Сжатие объектов: 100% (30/30), готово.
Запись объектов: 100% (38/38), 343.04 Киб | 2.54 Мб/с, готово.
Всего 38 (изменений 4), повторно использовано 0 (изменений 0), повторно использовано пакетов 0
remote: Resolving deltas: 100% (4/4), completed with 1 local object.
To github.com:eosihchev/study_2022-2023_os-intro.git
 74e5e32..8b1c2bf master -> master
[eosihchev@fedora os-intro]$
```

**eosihchev / study\_2022-2023\_os-intro** Public  
generated from yamadharma/course-directory-student-template

[Code](#) [Issues](#) [Pull requests](#) [Actions](#) [Projects](#) [Wiki](#) [Security](#) [Insights](#) [Settings](#)

master 1 branch 0 tags [Go to file](#) [Add file](#) [Code](#)

eosihchev feat(main): make course structure 8b1c2bf yesterday 2 commits		
config	Initial commit	yesterday
labs	feat(main): make course structure	yesterday
presentation	feat(main): make course structure	yesterday
project-personal	feat(main): make course structure	yesterday
template	Initial commit	yesterday
.gitattributes	Initial commit	yesterday
.gitignore	Initial commit	yesterday
.gitmodules	Initial commit	yesterday
CHANGELOG.md	Initial commit	yesterday
COURSE	feat(main): make course structure	yesterday
LICENSE	Initial commit	yesterday
Makefile	Initial commit	yesterday
README.en.md	Initial commit	yesterday
README.git-flow.md	Initial commit	yesterday
README.md	Initial commit	yesterday
prepare	feat(main): make course structure	yesterday

**About**

No description, website, or topics provided.

[Readme](#)  
CC-BY-4.0 license  
0 stars  
1 watching  
0 forks

**Releases**

No releases published  
[Create a new release](#)

**Packages**

No packages published  
[Publish your first package](#)

**Languages**

Python 93.6%	TeX 5.2%
Makefile 1.1%	Shell 0.1%

### 3 Контрольные вопросы

1. Что такое системы контроля версий (VCS) и для решения каких задач они предназначаются?
2. Объясните следующие понятия VCS и их отношения: хранилище, commit, история, рабочая копия.

Хранилище (repository, сокр. repo), или репозиторий, — место хранения всех версий и служебной информации. • Коммит (commit; редко переводится как «слепок») — 1) синоним версии; 2) создание новой версии («сделать коммит», «закоммитить»). • Рабочая копия (working copy или working tree) — текущее состояние файлов проекта, основанное на версии из хранилища (обычно на последней)

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

Централизованные системы — это системы, которые используют архитектуру клиент / сервер, где один или несколько клиентских узлов напрямую подключены к центральному серверу. Пример - Wikipedia. В децентрализованных системах каждый узел принимает свое собственное решение. Конечное поведение системы является совокупностью решений отдельных узлов. Пример — Bitcoin.

В классических системах контроля версий используется централизованная модель, предполагающая наличие единого репозитория для хранения файлов. Выполнение большинства функций по управлению версиями осуществляется специальным сервером

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

Создадим локальный репозиторий. Сначала сделаем предварительную конфигурацию, указав имя и email владельца репозитория: `git config --global user.name "Имя Фамилия"` `git config --global user.email "work@mail"` и настроив utf-8 в выводе сообщений `git: git config --global quotepath false` Для инициализации локального репозитория, расположенного, например, в каталоге `~/tutorial`, необходимо ввести в командной строке: `cd mkdir tutorial cd tutorial git init`

5. Опишите порядок работы с общим хранилищем VCS.

Для последующей идентификации пользователя на сервере репозитория необходимо сгенерировать пару ключей (приватный и открытый): `ssh-keygen -C "Имя Фамилия work@mail"` Ключи сохраняются в каталоге `~/.ssh/`. Скопировав из локальной консоли ключ в буфер обмена `cat ~/.ssh/id_rsa.pub | xclip -sel clip` вставляем ключ в появившееся на сайте поле.

6. Каковы основные задачи, решаемые инструментальным средством git?

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.

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 втекущей директории:`git status`–просмотртекущих изменений: `git diff`–сохранениетекущих 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 стекущим 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.

Использования `git` при работе с локальными репозиториями (добавления текстового документа в локальный репозиторий): `git add hello.txt` `git commit -am 'Новый файл'`

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

Ветки очень облегчают работу. Они решают такие проблемы как: нужно постоянно создавать архивы с рабочим кодом сложно “переключаться” между архивами сложно перетаскивать изменения между архивами легко что-то напутать или потерять

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. Для этого сначала нужно получить список имеющихся шаблонов: `curl -L -s https://www.gitignore.io/api/list` Затем скачать шаблон, например, для С и С++ `curl -L -s https://www.gitignore.io/api/c` » `.gitignore` `curl -L -s https://www.gitignore.io/api/c++` » `.gitignore`

## 4 Вывод

Я изучил идеологию и применение средств контроля версий. И освоил умения по работе с git.