
Front matter

lang: ru-RU

title: Лабораторная работа №2

subtitle: Операционные системы

author:

- Цоппа Е. Э.
institute:
- Российский университет дружбы народов, Москва, Россия
date: 19 февраля 2024

i18n babel

babel-lang: russian

babel-otherlangs: english

Formatting pdf

toc: false

toc-title: Содержание

slide_level: 2

aspectratio: 169

section-titles: true

theme: metropolis

header-includes:

- \metroset{progressbar=frametitle,sectionpage=progressbar,numbering=fraction}
- '\makeatletter'
- '\beamer@ignorenonframefalse'
- '\makeatother'

Цель работы

Цель данной лабораторной работы -- изучение идеологии и применения средств контроля версий, освоение умения по работе с git.

Задание

1. Создать базовую конфигурацию для работы с git
2. Создать ключ SSH
3. Создать ключ GPG
4. Настроить подписи Git

5. Зарегистрироваться на GitHub
6. Создать локальный каталог для выполнения заданий по предмету.

Выполнение лабораторной работы

Установка программного обеспечения

Устанавливаю необходимое программное обеспечение git и gh через терминал с помощью команд: `dnf install git` и `dnf install gh`

```
[evatsoppa@fedora ~]$ sudo dnf -y install git
[sudo] пароль для evatsoppa:
Последняя проверка окончания срока действия метаданных: 2:30:34 назад, Вс 18 фев 2024 12:09:43
.
Пакет git-2.40.0-1.fc38.x86_64 уже установлен.
Зависимости разрешены.
Нет действий для выполнения.
Выполнено!
[evatsoppa@fedora ~]$ sudo dnf -y install gh
Последняя проверка окончания срока действия метаданных: 2:30:55 назад, Вс 18 фев 2024 12:09:43
.
Зависимости разрешены.
=====
Пакет          Архитектура      Версия           Репозиторий      Размер
=====
Установка:
gh              x86_64           2.36.0-1.fc38    updates          8.9 М
```

Базовая настройка git

Задаю в качестве имени и email владельца репозитория свои имя, фамилию и электронную почту

```
[evatsoppa@fedora ~]$ git config --global user.name "Eva Tsoppa"
[evatsoppa@fedora ~]$ git config --global user.email "1132236045@pfur.ru"
```

Настраиваю utf-8 в выводе сообщений git для их корректного отображения

```
[evatsoppa@fedora ~]$ git config --global core.quotepath false
```

Начальной ветке задаю имя master

```
[evatsoppa@fedora ~]$ git config --global init.defaultBranch master
```

Задаю параметры autocrlf и safecrlf

```
[evatsoppa@fedora ~]$ git config --global core.autocrlf input
[evatsoppa@fedora ~]$ git config --global core.safecrlf warn
[evatsoppa@fedora ~]$
```

Создание ключа SSH

Создаю ключ ssh размером 4096 бит по алгоритму rsa

```
[evatsoppa@fedora ~]$ ssh-keygen -t rsa -b 4096
Generating public/private rsa key pair.
Enter file in which to save the key (/home/evatsoppa/.ssh/id_rsa):
Created directory '/home/evatsoppa/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/evatsoppa/.ssh/id_rsa
Your public key has been saved in /home/evatsoppa/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:pQG8+Ixwo0SLGfg8BTg+rFNcgJg0hTnWBAgQThAIX0E evatsoppa@fedora
The key's randomart image is:
+---[RSA 4096]-----+
| ^&XBo.. |
| ^oo=o .. |
| *OoE . . . |
| =O + . + |
| .o.= = S |
| o . . o |
| . |
| |
| |
+-----[SHA256]-----+
```

Создаю ключ ssh по алгоритму ed25519

```
[evatsoppa@fedora ~]$ ssh-keygen -t ed25519
Generating public/private ed25519 key pair.
Enter file in which to save the key (/home/evatsoppa/.ssh/id_ed25519):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/evatsoppa/.ssh/id_ed25519
Your public key has been saved in /home/evatsoppa/.ssh/id_ed25519.pub
The key fingerprint is:
SHA256:jHlXRJR1lbZsRHntAzSnEfsqUZ4/ywFKIdIpjEfz+eY evatsoppa@fedora
The key's randomart image is:
+---[ED25519 256]--+
|      o  .==oo=|
|    + + o= o**o|
|  . = *o..+*.o|
|    .+o o.+ o=.|
|    o S . = +...|
|    . .+ o + |
|      E . + |
|      . . + |
|      o |
+-----[SHA256]-----+
```

Создание ключа GPG

Генерирую ключ GPG, затем выбираю тип ключа RSA and RSA, задаю максимальную длину ключа: 4096, оставляю неограниченный срок действия ключа. Далее отвечаю на вопросы программы о личной информации

```
[evatsoppa@fedora ~]$ gpg --full-generate-key
gpg (GnuPG) 2.4.0; 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.

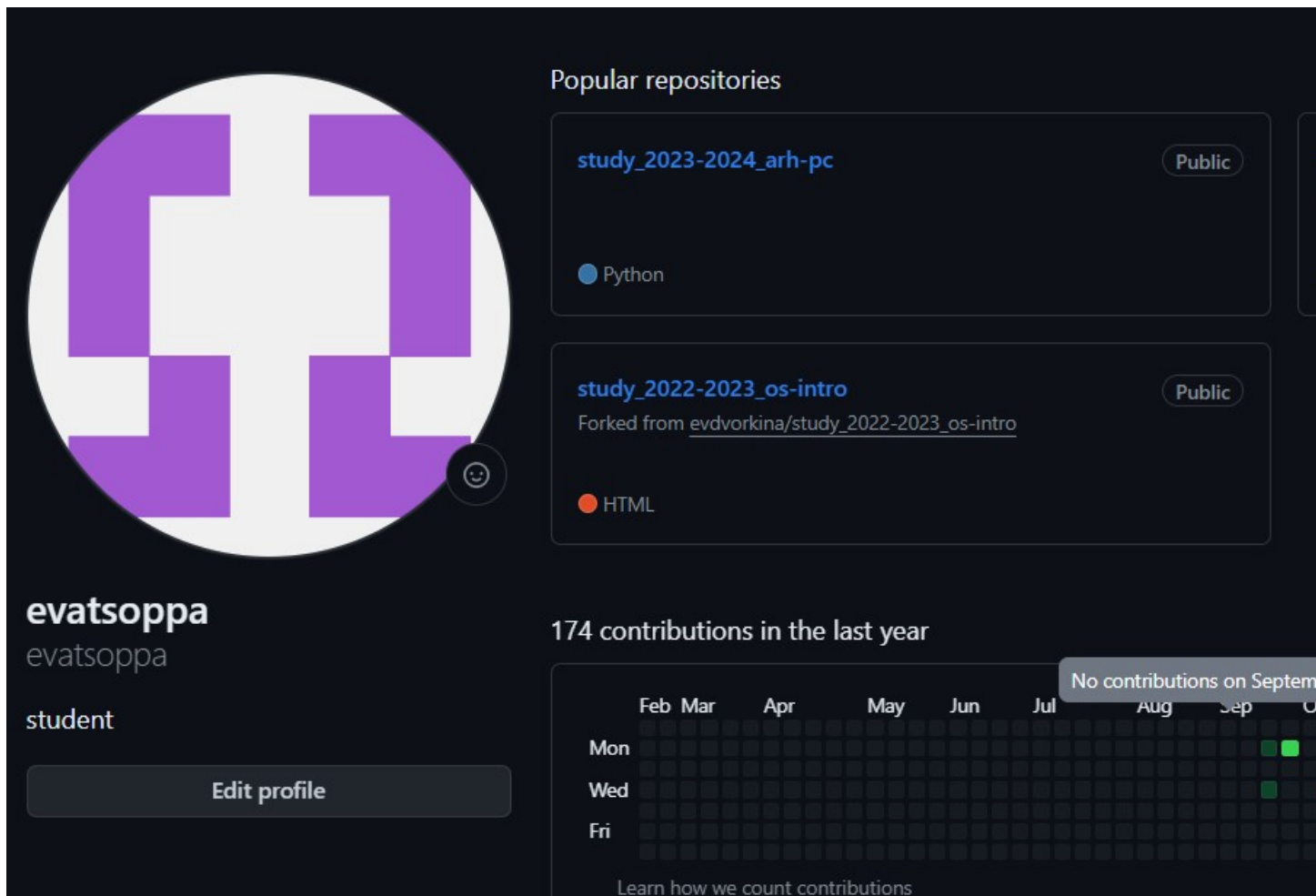
gpg: создан каталог '/home/evatsoppa/.gnupg'
gpg: создан щит с ключами '/home/evatsoppa/.gnupg/pubring.kbx'
Выберите тип ключа:
  (1) RSA and RSA
  (2) DSA and Elgamal
  (3) DSA (sign only)
  (4) RSA (sign only)
  (9) ECC (sign and encrypt) *default*
 (10) ECC (только для подписи)
 (14) Existing key 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: TsoppaEva
Адрес электронной почты: 1132236045@pfur.ru
```

Регистрация на Github

У меня уже был создан аккаунт на Github, соответственно, основные данные аккаунта я так же заполняла и проводила его настройку, поэтому просто вхожу в свой аккаунт



The screenshot shows the GitHub profile of a user named 'evatsoppa'. The profile picture is a circular avatar with a purple and white geometric pattern. The username 'evatsoppa' is displayed in a large font, with 'evatsoppa' in a smaller font below it. The bio 'student' is shown. There is an 'Edit profile' button. The 'Popular repositories' section lists two repositories: 'study_2023-2024_arh-pc' (Python) and 'study_2022-2023_os-intro' (HTML). The '174 contributions in the last year' section shows a calendar grid with green squares indicating contributions. A tooltip indicates 'No contributions on September 1st'.

Добавление ключа GPG в Github

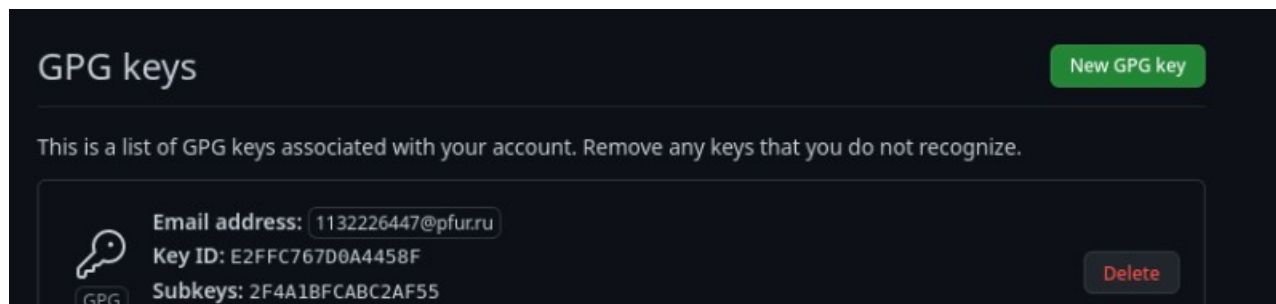
Вывожу список созданных ключей в терминал, ищу в результате запроса отпечаток ключа (последовательность байтов для идентификации более длинного, по сравнению с самим отпечатком, ключа), он стоит после знака следа, копирую его в буфер обмена

```
[evatsoppa@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, 1u
/home/evatsoppa/.gnupg/pubring.kbx
-----
sec   rsa4096/728E8725898C0623 2024-02-18 [SC]
      15C86E2C3C65D488A7FCD0CF3728E8725898C0623
uid           [ абсолютно ] TsoppaEva (собака) <1132236045@pfur.ru>
ssb   rsa4096/1610F6ED2A698994 2024-02-18 [E]
```

Ввожу в терминале команду, с помощью которой копирую сам ключ GPG в буфер обмена

```
[evatsoppa@fedora ~]$ gpg --armor --export E2FFC767D0A4458F | xclip -sel clip
```

Открываю настройки GitHub, ищу среди них добавление GPG ключа
Нажимаю на "New GPG key" и вставляю в поле ключ из буфера обмена
Я добавила ключ GPG на GitHub



Настроить подписи Git

Настраиваю автоматические подписи коммитов git: используя введенный ранее email, указываю git использовать его при создании подписей коммитов

```
[evatsoppa@fedora ~]$ git config --global user.signingkey 728E8725898C0623
[evatsoppa@fedora ~]$ git config --global commit.gpgsign true
[evatsoppa@fedora ~]$ git config --global gpg.program $(which gpg2)
```

Настройка gh

Начинаю авторизацию в gh, отвечаю на наводящие вопросы от утилиты, в конце выбираю авторизоваться через браузер

```
? 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
```

Вижу сообщение о завершении авторизации под именем evatsoppa

```
✓ Authentication complete.
- gh config set -h github.com git_protocol https
✓ Configured git protocol
✓ Logged in as evatsoppa
```

Создание репозитория курса на основе шаблона

Сначала создаю директорию с помощью утилиты mkdir

Перехожу в только что созданную директорию "Операционные системы".

Далее в терминале ввожу команду `gh repo create study_2023-2024_os-intro --template yamadharm/course-directory-student-trmplate --public`

Клонирую репозиторий к себе в директорию

```
[evatsoppa@fedora ~]$ mkdir -p ~/work/study/2023-2024/"Операционные системы"
[evatsoppa@fedora ~]$ cd ~/work/study/2023-2024/"Операционные системы"
[evatsoppa@fedora Операционные системы]$ gh repo create study_2023-2024_os-intro --template=yamadharma/course-di
t-template --public
✓ Created repository evatsoppa/study_2023-2024_os-intro on GitHub
[evatsoppa@fedora Операционные системы]$ git clone --recursive https://github.com/evatsoppa/study_2023-2024_os-i
tro
Клонирование в «os-intro»...
```

Перехожу в каталог курса

Удаляю лишние файлы с помощью утилиты rm, далее создаю необходимые каталоги используя makefile

Добавляю все новые файлы для отправки на сервер (сохраняю добавленные изменения) с помощью команды git add и комментирую их с помощью git commit

```
[evatsoppa@fedora os-intro]$ git add .
[evatsoppa@fedora os-intro]$ git commit -am 'feat(main): make course structure'
git: «commit» не является командой git. Смотрите «git --help».

Самые похожие команды:
  commit
[evatsoppa@fedora os-intro]$ git commit -am 'feat(main): make course structure'
[master c228868] feat(main): make course structure
 2 files changed, 1 insertion(+), 14 deletions(-)
 delete mode 100644 package.json
```

Отправляю файлы на сервер с помощью git push

```
[evatsoppa@fedora os-intro]$ git push
Перечисление объектов: 5, готово.
Подсчет объектов: 100% (5/5), готово.
При сжатии изменений используется до 5 потоков
Сжатие объектов: 100% (2/2), готово.
Запись объектов: 100% (3/3), 950 байтов | 950.00 КиБ/с, готово.
Всего 3 (изменений 1), повторно использовано 0 (изменений 0), повторно использовано пакетов 0
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To https://github.com/evatsoppa/study_2023-2024_os-intro.git
 0f325e4..c228868 master -> master
```

Выводы

При выполнении данной лабораторной работы я изучила идеологию и применение средств контроля версий, освоила умение по работе с git.

Ответы на контрольные вопросы.

1. Системы контроля версий (VCS) - программное обеспечение для облегчения работы с изменяющейся информацией.
Они позволяют хранить несколько версий изменяющейся информации, одного и того же документа, может предоставить доступ к более ранним версиям документа.
Используется для работы нескольких человек над проектом, позволяет посмотреть, кто и когда внес какое-либо изменение и т. д. VCS применяются для: Хранения понлой истории изменений,

сохранения причин всех изменений, поиска причин изменений и совершивших изменение, совместной работы над проектами.

2. Хранилище -- репозиторий, хранилище версий, в нем хранятся все документы, включая историю их изменения и прочей служебной информацией.
commit -- отслеживание изменений, сохраняет разницу в изменениях.
История -- хранит все изменения в проекте и позволяет при необходимости вернуться/обратиться к нужным данным.
Рабочая копия -- копия проекта, основанная на версии из хранилища, чаще всего последней версии.
3. Централизованные VCS (например: CVS, TFS, AccuRev) -- одно основное хранилище всего проекта. Каждый пользователь копирует себе необходимые ему файлы из этого репозитория, изменяет, затем добавляет изменения обратно в хранилище. Децентрализованные VCS (например: Git, Bazaar) -- у каждого пользователя свой вариант репозитория (возможно несколько вариантов), есть возможность добавлять и забирать изменения из любого репозитория. В отличие от классических, в распределенных (децентрализованных) системах контроля версий центральный репозиторий не является обязательным.
4. Сначала создается и подключается удаленный репозиторий, затем по мере изменения проекта эти изменения отправляются на сервер.
5. Участник проекта перед началом работы получает нужную ему версию проекта в хранилище, с помощью определенных команд, после внесения изменений пользователь размещает новую версию в хранилище. При этом предыдущие версии не удаляются. К ним можно вернуться в любой момент.
6. Хранение информации о всех изменениях в вашем коде, обеспечение удобства командной работы над кодом.
7. 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. `git push -all` отправляем из локального репозитория все сохраненные изменения в центральный репозиторий, предварительно создав локальный репозиторий и сделав предварительную конфигурацию.
9. Ветвление - один из параллельных участков в одном хранилище, исходящих из одной версии, обычно есть главная ветка.
Между ветками, т. е. их концами возможно их слияние. Используются для разработки новых функций.
10. Во время работы над проектом могут создаваться файлы, которые не следуют добавлять в репозиторий. Например, временные файлы.
Можно прописать шаблоны игнорируемых при добавлении в репозиторий типов файлов в файл `.gitignore` с помощью сервисов.

Список литературы

1. Лабораторная работа № 2