Отчет по лабораторной работе №2

Дисциплина архитектура компьютера

Никуленков Степан Сергеевич

Содержание

# 1 Цель работы

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

# 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.

# 3 Теоретическое введение

Системы контроля версий. Общие понятия

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 дельта-компрессию — 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.

# 4 Выполнение лабораторной работы

Установливаю git: dnf install git

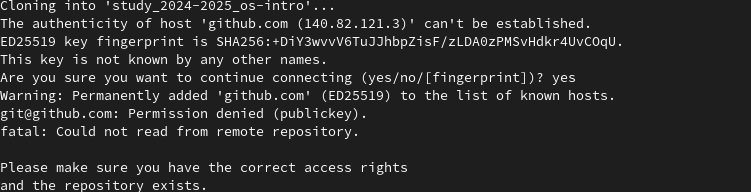


Рис. 1: установка

Устанавливаю gh: dnf install gh

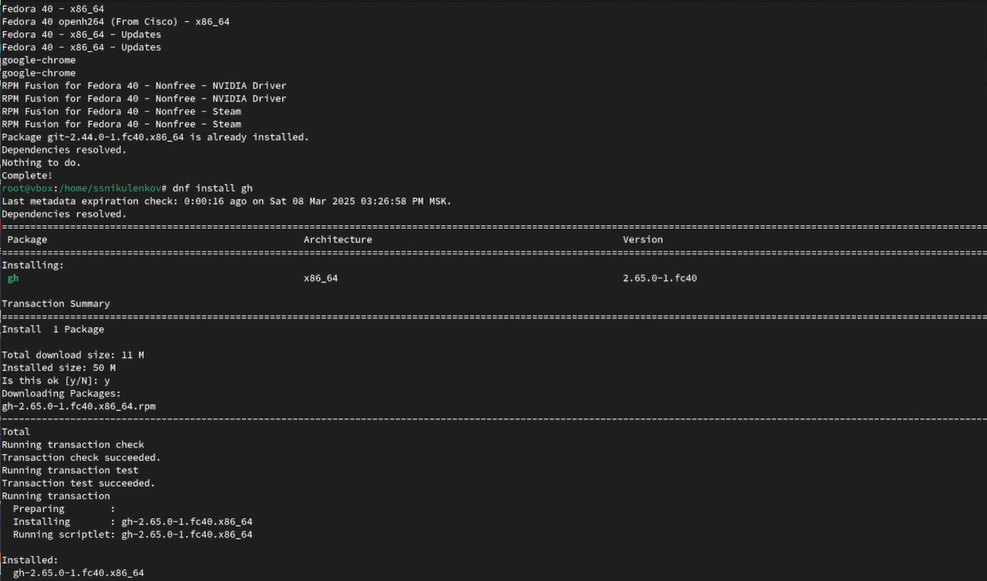


Рис. 2: установка

Базовая настройка git

Зададаю имя и email для моего репозитория:

git config –global user.name “Stepan Nikulenkov” git config –global user.email “stepannikulenkov@gmail.com”

Настраиваю utf-8 в выводе сообщений git:

git config –global core.quotepath false

Настраиваю верификацию и подписание коммитов git

Зададаю имя начальной ветки (будем называть её master):

git config –global init.defaultBranch master

Параметр autocrlf:

git config –global core.autocrlf input

Параметр safecrlf:

git config –global core.safecrlf warn

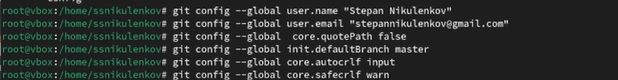


Рис. 3: базовая настройка git

Создаю ключи ssh по алгоритму rsa с ключём размером 4096 бит:

ssh-keygen -t rsa -b 4096

по алгоритму ed25519:

ssh-keygen -t ed25519

Создаю ключи pgp

Генерируем ключ gpg –full-generate-key

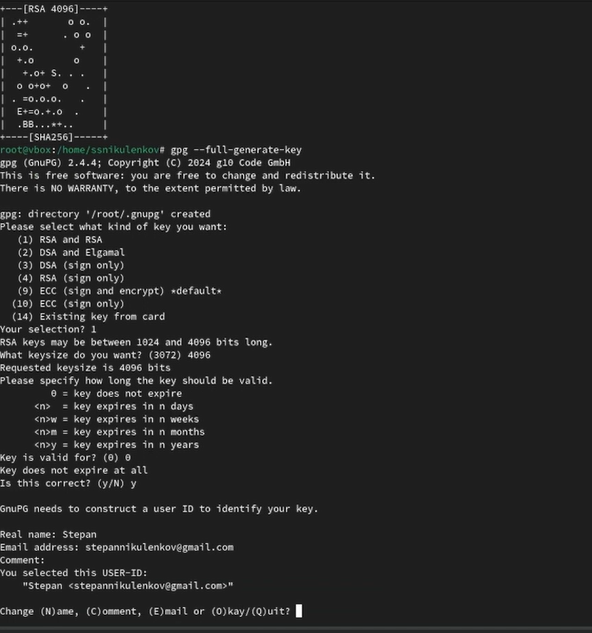


Рис. 4: создание ключей



Рис. 5: окно подтверждения

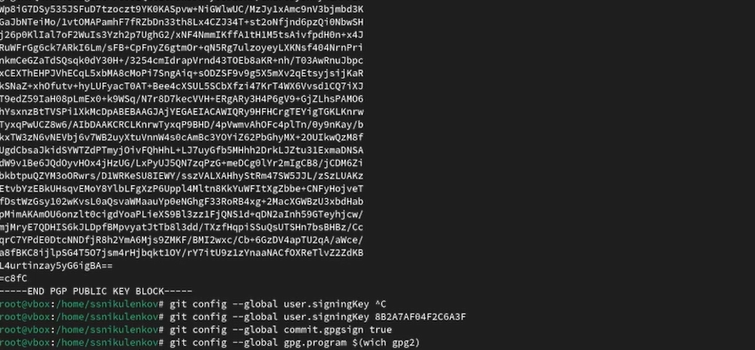


Рис. 6: ключ

Выводим список ключей и копируем отпечаток приватного ключа.Cкопировал сгенерированный PGP ключ в буфер обмена:

gpg –armor –export | xclip -sel clip

Перехожу в настройки GitHub , нажмимаю на кнопку New GPG key и вставьте полученный ключ в поле ввода.

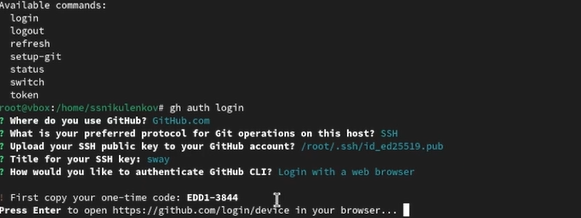


Рис. 7: копирование ключа

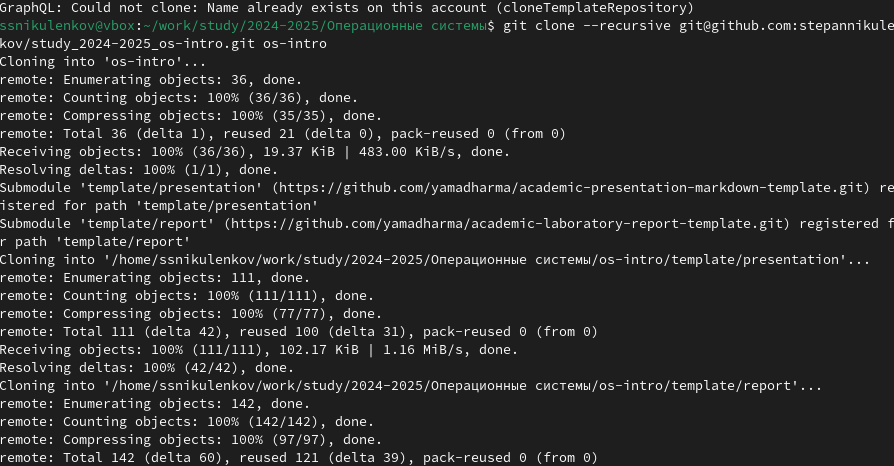


Рис. 8: добавление ключа

Авторизуюсь через браузер создание репозитория курса на основе шаблона прописываю комнды для создания репозитория:

mkdir -p ~/work/study/2024-2025/“Операционные системы” cd ~/work/study/2024-2025/“Операционные системы” gh repo create study\_2024-2025\_os-intro –template=yamadharma/course-directory-student-template –public git clone –recursive git@github.com:/study\_2024-2025\_os-intro.git os-intro

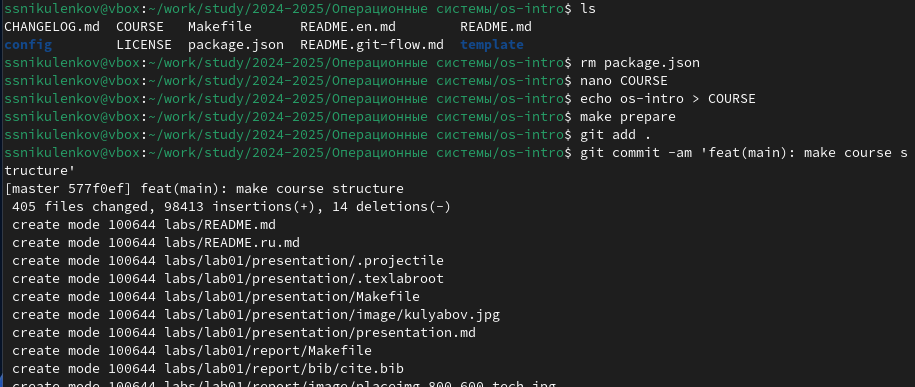


Рис. 9: создание репозитория

Настройка каталога курса

Перехожу в каталог курса:

cd ~/work/study/2024-2025/“Операционные системы”/os-intro

Удаляю лишние файлы:

rm package.json

Создаю необходимые каталоги:

echo os-intro > COURSE make

Отправляю файлы на сервер:

git add . git commit -am ‘feat(main): make course structure’ git push

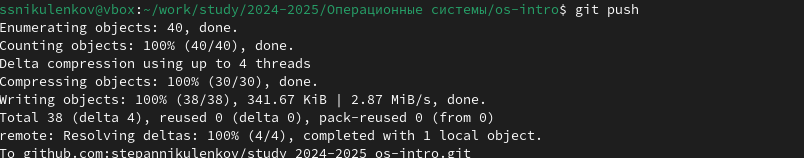


Рис. 10: отправка на сервер

# 5 Выводы

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