Лабораторная работа №2

Первоначальная настройка git

Комягин Андрей Николаевич

Содержание

# 1 Цель работыЙ

Изучить концепцию и применение средств контроля версий. Приобрести навыки работы с git.

# 2 Задание

* Создать базовую конфигурацию для работы с git
* Создать ключ SSH
* Создать ключ PGP
* Настроить подписи git
* Зарегистрироваться на Github
* Создать локальный каталог для выполнения заданий по предмету

# 3 Выполнение лабораторной работы

Установим необходимое ПО.(рис. 1).



Рис. 1: Установка ПО

Настроим git. Зададим имя и почту владельца репозитория, настроим utf-8, зададим имя начальной ветки и настроим параметры отступов рис. (рис. 2).

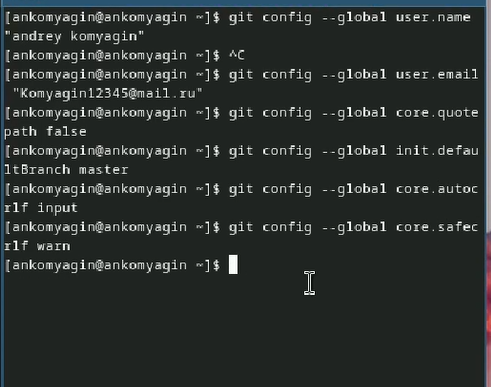


Рис. 2: Первичная настройка git

Далее создадим ssh ключи по двум разным алгоритмам - rsa(4096)(рис. 3) и ed25519(рис. 4).

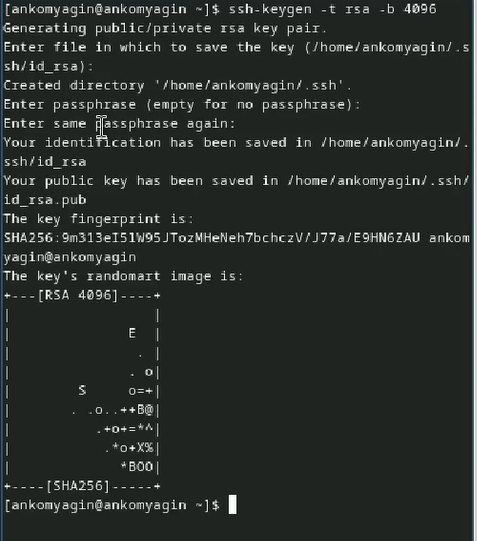


Рис. 3: алгоритм rsa

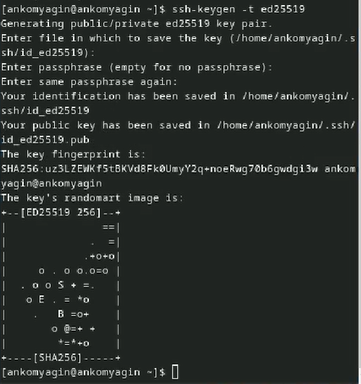


Рис. 4: алгоритм ed25519

Также создадим ключ gpg с необходимыми опциями (рис. 5).

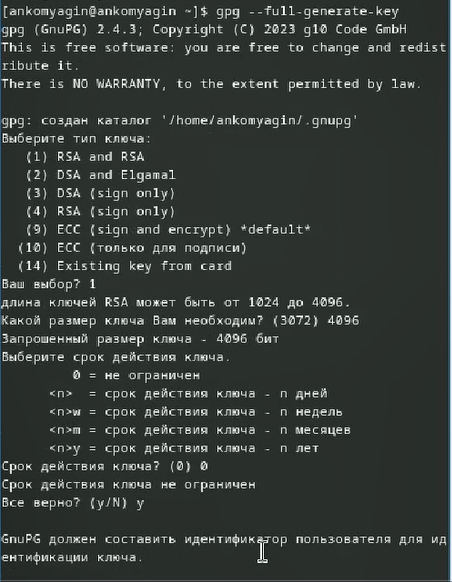


Рис. 5: ключ gpg

Далее нееобходимо создать учётную запись на Github. Я пропускаю данный этап, так как Github имеется с первого семестра. Выведем список ключей и скопируем отпечаток приватного ключа. Затем ключ в буфер обмена (рис. 6)

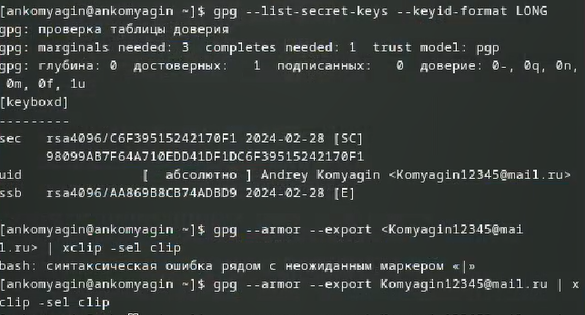


Рис. 6: Копирование pgp ключа

Добавим скопированный ключ на Github (рис. 7)

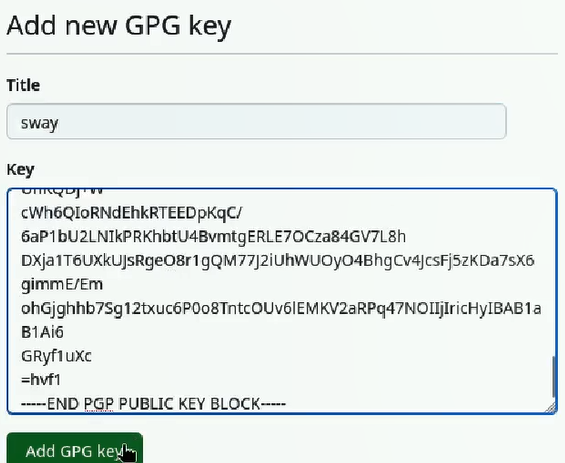


Рис. 7: Добавление ключа на Github

Настроим автоматические подписи коммитов git (рис. 8)

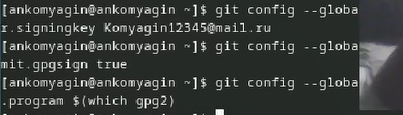


Рис. 8: Настройка подписей

Затем настроим gh и авторизуемся (рис. 9)

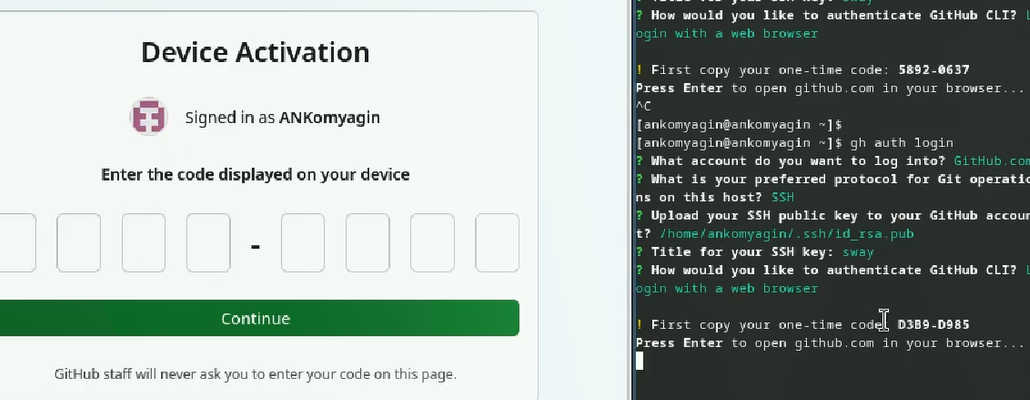


Рис. 9: gh авторизация

Создадим репозиторий курса на основе шаблона (рис. 10)

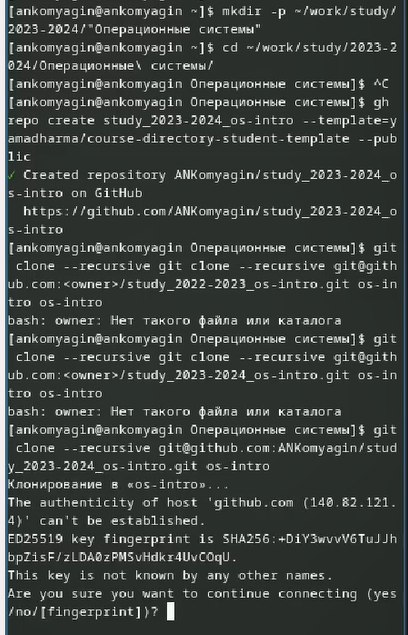


Рис. 10: создание репозитория

Настроим каталог курса, а затем отправим его на Github (рис. 11)

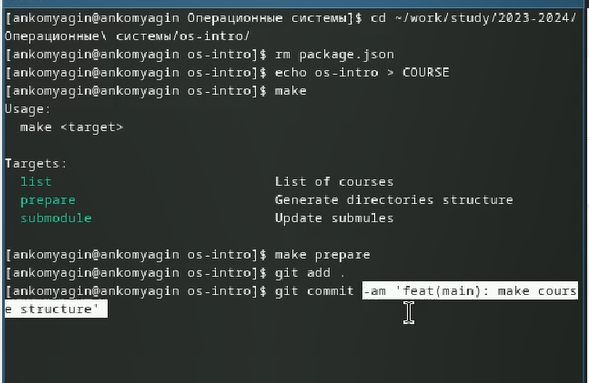


Рис. 11: Настройка каталога

# 4 Контрольные вопросы

1. Что такое системы контроля версий (VCS) и для решения каких задач они предназначаются? Системы контроля версий (VCS) - это инструменты, которые помогают отслеживать изменения в исходном коде и управлять ими. Они предназначены для решения задач хранения истории изменений, совместной работы над проектами, отката к предыдущим версиям и т. д.
2. Объясните следующие понятия VCS и их отношения: хранилище, commit, история, рабочая копия.

* Хранилище (repository) - это место, где хранится история изменений проекта.
* Commit - это операция сохранения изменений в репозитории.
* История (history) - это список всех коммитов, которые были сделаны в проекте.
* Рабочая копия (working copy) - это каталог на компьютере разработчика, в котором он работает над проектом.

1. Что представляют собой и чем отличаются централизованные и децентрализованные VCS? Централизованные VCS имеют единственный центральный сервер, к которому подключаются все разработчики (например, SVN). Децентрализованные VCS позволяют каждому разработчику иметь полную копию репозитория, с которой он может работать независимо (например, Git).
2. Опишите действия с VCS при единоличной работе с хранилищем. При единоличной работе с хранилищем в VCS разработчик делает изменения в своей рабочей копии проекта, коммитит их в локальный репозиторий и при необходимости откатывается к предыдущим версиям.
3. Опишите порядок работы с общим хранилищем VCS. Порядок работы с общим хранилищем VCS включает получение изменений из удаленного репозитория, коммит изменений в локальный репозиторий и отправку изменений обратно в удаленный репозиторий.
4. Каковы основные задачи, решаемые инструментальным средством git? Git предназначен для управления версиями файлов, совместной работы над проектами, отслеживания изменений и управления различными ветками разработки.
5. Назовите и дайте краткую характеристику командам git.

* git add - добавление файла в индекс (обычно точка).
* git commit -m “Сообщение” - создание коммита с описанием изменений.
* git push - отправка изменений в удаленный репозиторий.
* git pull - получение изменений из удаленного репозитория.

1. Приведите примеры использования при работе с локальным и удалённым репозиториями. При работе с локальным репозиторием можно использовать команды git add, git commit. При работе с удаленным репозиторием - git clone, git push, git pull.
2. Что такое и зачем могут быть нужны ветви (branches)? Ветви (branches) в Git позволяют разрабатывать различные функциональности независимо друг от друга, а затем объединять изменения.
3. Как и можно игнорировать некоторые файлы при commit? Для игнорирования некоторых файлов при коммите можно использовать файл .gitignore, в который записываются шаблоны файлов или папок, которые не должны попадать в репозиторий.

# 5 Выводы

В ходе выполнения лабораторной работы я повторил правила работы с git, узнал о системе подписей и pgp ключах.

# Список литературы

[Туис, курс Архитектура компьютера и операционные системы](https://esystem.rudn.ru/course/view.php?id=5790)