**РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ**

**Факультет физико-математических и естественных наук**

**Кафедра прикладной информатики и теории вероятностей**

**ОТЧЕТ  
ПО ЛАБОРАТОРНОЙ РАБОТЕ №2**

*дисциплина: Операционные системы*

**Студент:** Степанов Иван

**Группа:** НБИбд-03-21

**Ст. билет №:** 1032213710

Москва

2022 г.

**Цель работы:**

– Изучить идеологию и применение средств контроля версий.

– Освоить умения по работе с git.

**Ход работы:**

Сначала надо было создать учетную запись и заполнить её, учетная запись у меня была создана до лабораторной работы, так что это пункт уже был выполнен.

Далее я сделал права суперпользователя на линуксе, чтобы установить git-flow, после чего я установил git-flow и gh (рис. 1.1, 1.2)

После я совершил базовую настройку гит (рис. 2.1)

Далее я создал ключ ssh (рис. 3.1)

после этого я создал ключ pgp и добавил его на гитхаб (рис. 4.1, 4.2, 4.3, 4.4, рис 4.5)

далее я настроил автоматические подписи коммитов гит (рис. 5.1)

после чего я авторизовался с помощью команды “gh auth login” (рис 6.1)

потом с помощью шаблона рабочего пространства я создал репозиторий на своем гитхабе (рис. 7.1, 7.2)

далее я настроил каталог курса: удалил лишние файлы, создал необходимые каталоги, и отправил файлы на сервер. (рис. 8.1)

**контрольные вопросы:**

**1.** 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.

2. коммит это контрольные точки, то есть пакет изменений хранящий информацию с добавленными отредактированными или удаленными файлами кода.

Хранилище — это база данных, содержащая папки и файлы

История это история того как изменялся проект по разным временным меткам, чтобы было удобно просматривать изменения в проекте.

Рабочая копия это последняя версия рабочего проекта.

3. 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. 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](mailto: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

6. 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. Основные команды 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 изменения: 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 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. Использования git при работе с локальными репозиториями (добавления текстового документа в локальный репозиторий):

git add hello.txt

git commit -am 'Новый файл’

git push

9. Проблемы, которые решают ветки git:

* нужно постоянно создавать архивы с рабочим кодом
* сложно "переключаться" между архивами
* сложно перетаскивать изменения между архивами
* легко что-то напутать или потерять

10. 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.

Вывод: я узнал про VCS, а так же на практике получил навыки по работе с git.

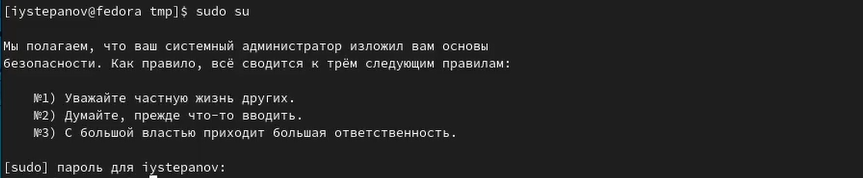


Рис. 1.1

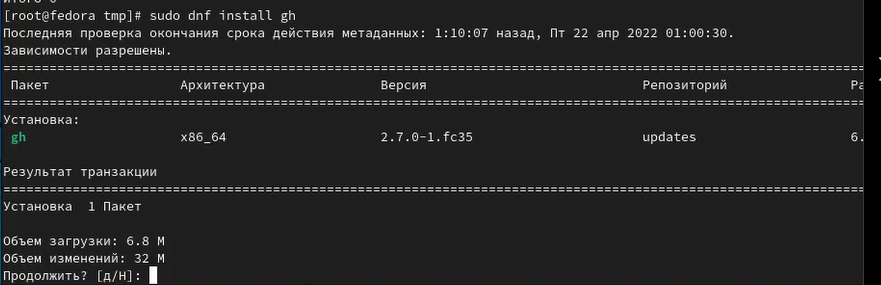


Рис. 1.2

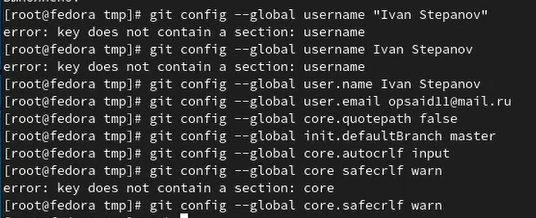


Рис 2.1

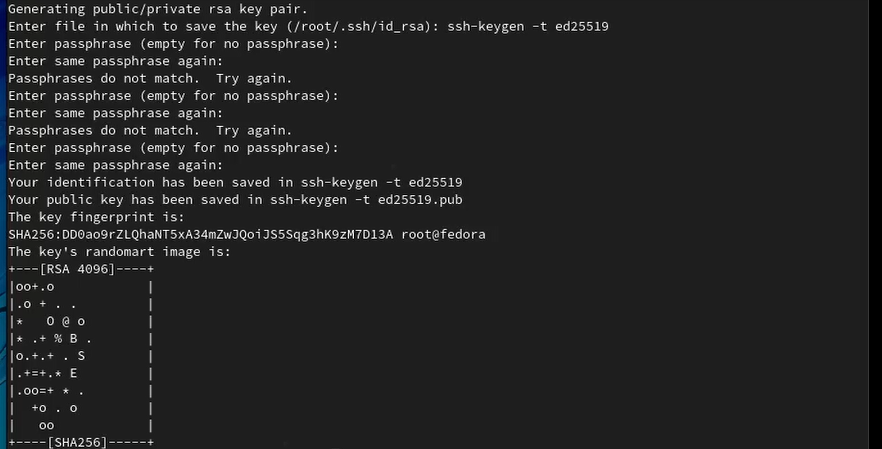


Рис 3.1

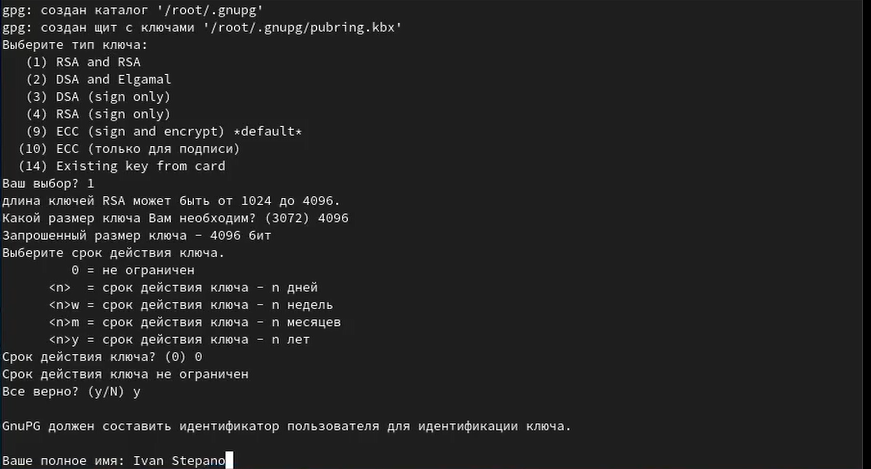
рис 4.1

рис 4.2

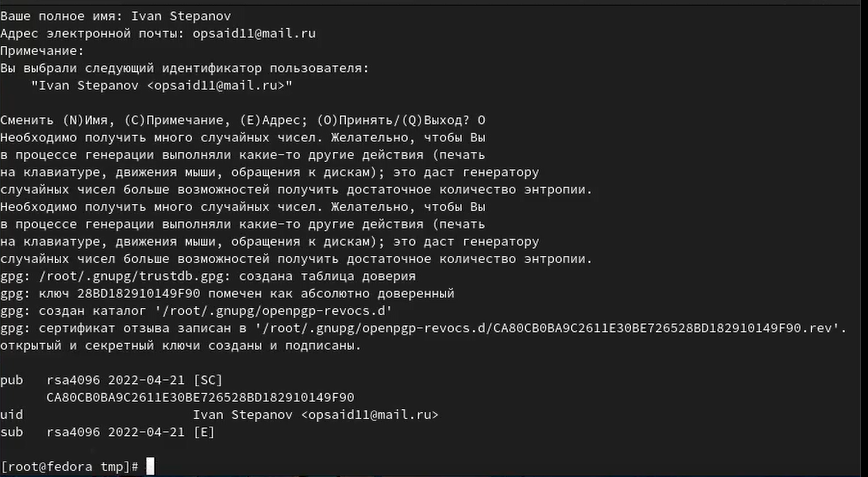


Рис 4.3

Рис. 4.4

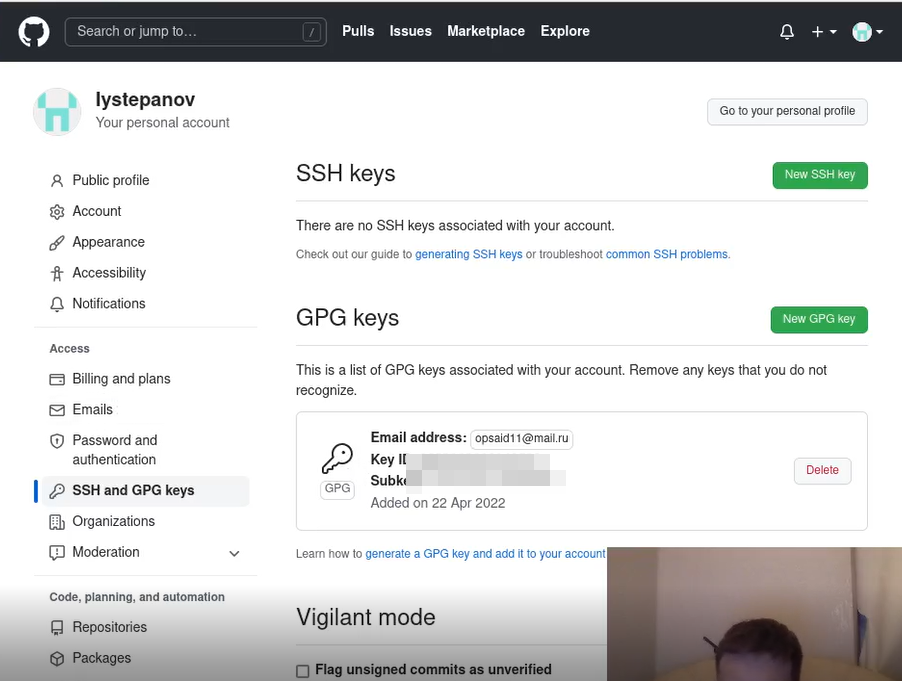


Рис 4.5

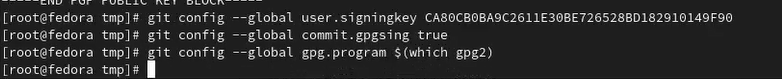


Рис. 5.1

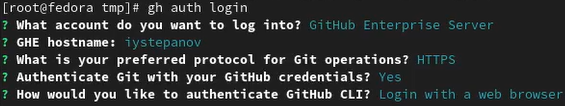


Рис 6.1

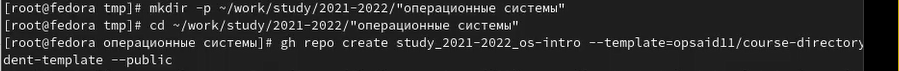


Рис. 7.1

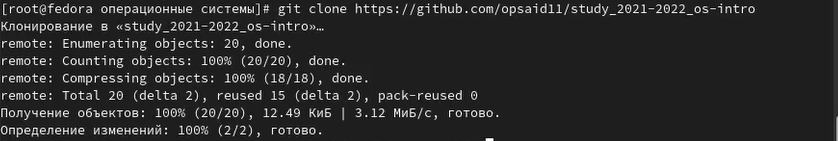


Рис. 7.2



Рис. 8.1

Вывод: я узнал про VCS, а так же на практике получил навыки по работе с git.