

# **Отчёт по лабораторной работе №2**

**Дисциплина: Операционные системы**

Филатов Илья Гурамович

# Содержание

<b>1</b>	<b>Цель работы</b>	<b>4</b>
<b>2</b>	<b>Задание</b>	<b>5</b>
<b>3</b>	<b>Выполнение лабораторной работы</b>	<b>6</b>
3.1	Установка программного обеспечения . . . . .	6
3.2	Создание SSH ключей . . . . .	7
3.3	Создание PGP ключа . . . . .	7
3.4	Добавление PGP ключа в GitHub . . . . .	8
3.5	Настройка автоматических подписей коммитов git . . . . .	9
3.6	Настройка gh . . . . .	9
3.7	Создание репозитория курса на основе шаблона . . . . .	10
3.8	Ответы на контрольные вопросы . . . . .	11
<b>4</b>	<b>Выводы</b>	<b>13</b>
<b>5</b>	<b>Список литературы</b>	<b>14</b>

# Список иллюстраций

3.1	Установка пакетов . . . . .	6
3.2	Установка базовых параметров . . . . .	7
3.3	Создание ключа SSH . . . . .	7
3.4	Создание PGP ключа . . . . .	8
3.5	Настройка PGP ключа в github . . . . .	8
3.6	PGP авторизация . . . . .	9
3.7	Авторизация gh . . . . .	9
3.8	Создание собственного репозитория . . . . .	10
3.9	Настройка каталога курса . . . . .	10

# 1 Цель работы

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

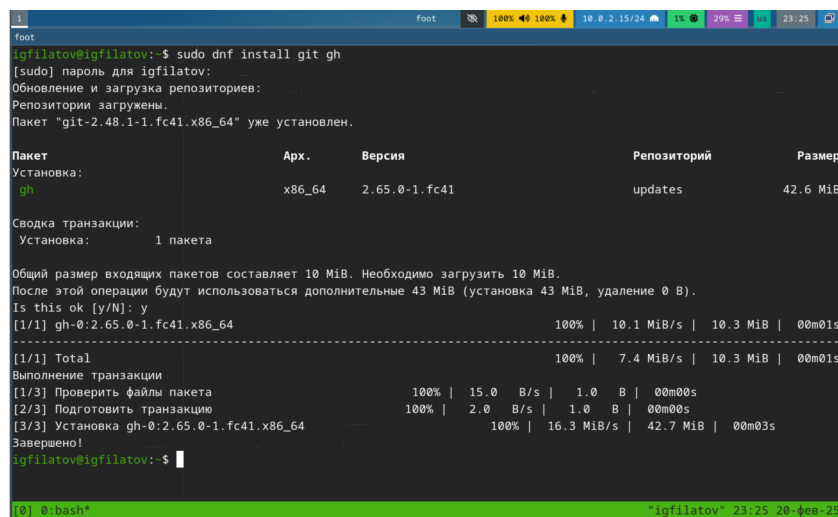
## 2 Задание

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

## 3 Выполнение лабораторной работы

### 3.1 Установка программного обеспечения

Установим git и gh (рис. 3.1).



```
igfilatov@igfilatov: ~$ sudo dnf install git gh
[sudo] пароль для igfilatov:
Обновление и загрузка репозитория:
Репозитории загружены.
Пакет "git-2.48.1-1.fc41.x86_64" уже установлен.

Пакет
Установка:
  gh
Арх.
  x86_64
Версия
  2.65.0-1.fc41
Репозиторий
  updates
Размер
  42.6 MiB

Сводка транзакции:
  Установка: 1 пакета

Общий размер входящих пакетов составляет 10 MiB. Необходимо загрузить 10 MiB.
После этой операции будут использоваться дополнительные 43 MiB (установка 43 MiB, удаление 0 B).
Is this ok [y/N]: y
[1/1] gh-0:2.65.0-1.fc41.x86_64 100% | 10.1 MiB/s | 10.3 MiB | 00m01s
-----
[1/1] Total 100% | 7.4 MiB/s | 10.3 MiB | 00m01s
Выполнение транзакции
[1/3] Проверить файлы пакета 100% | 15.0 B/s | 1.0 B | 00m00s
[2/3] Подготовить транзакцию 100% | 2.0 B/s | 1.0 B | 00m00s
[3/3] Установка gh-0:2.65.0-1.fc41.x86_64 100% | 16.3 MiB/s | 42.7 MiB | 00m03s
Завершено!
igfilatov@igfilatov: ~$
```

Рис. 3.1: Установка пакетов

Зададим базовые параметры git, также пропустим шаг регистрации на GitHub поскольку это уже было выполнено (рис. 3.2).

```
1
foot
igfilatov@igfilatov:~$ git config --global user.name "Илья Филатов"
igfilatov@igfilatov:~$ git config --global user.email "ila689213@gmail.com"
igfilatov@igfilatov:~$ git config --global core.quotepath false
igfilatov@igfilatov:~$ git config --global init.defaultBranch master
igfilatov@igfilatov:~$ git config --global core.autocrlf input
igfilatov@igfilatov:~$ git config --global core.safecrlf warn
igfilatov@igfilatov:~$
```

Рис. 3.2: Установка базовых параметров

## 3.2 Создание SSH ключей

Создадим ключ SSH по алгоритму ed25519 (рис. 3.3).

```
1
foot
igfilatov@igfilatov:~$ ssh-keygen -t ed25519
Generating public/private ed25519 key pair.
Enter file in which to save the key (/home/igfilatov/.ssh/id_ed25519):
Created directory '/home/igfilatov/.ssh'.
Enter passphrase for '/home/igfilatov/.ssh/id_ed25519' (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/igfilatov/.ssh/id_ed25519
Your public key has been saved in /home/igfilatov/.ssh/id_ed25519.pub
The key fingerprint is:
SHA256:3xoNd0Tul7yMbn5rJj1BYG5MrmjRCEgmedLZKR7fhkc igfilatov@igfilatov
The key's randomart image is:
+--[ED25519 256]--+
| .+o+ . . |
| ooB + E o + |
| + + + + O . |
| . o B + = . |
| S + + . |
| + * + |
| . O + O + |
| O + O * |
| . + + + O |
+----[SHA256]-----+
igfilatov@igfilatov:~$
```

Рис. 3.3: Создание ключа SSH

## 3.3 Создание PGP ключа

Используя команду генерирую ключ и ввожу свои данные (рис. 3.4).

```
1
foot
Адрес электронной почты: ila689213@gmail.com
Примечание:
Используется таблица символов 'utf-8'.
Вы выбрали следующий идентификатор пользователя:
"Илья Филатов <ila689213@gmail.com>"

Сменить (N)Имя, (C)Примечание, (E)Адрес; (O)Принять/(Q)Выход? 0
Необходимо получить много случайных чисел. Желательно, чтобы Вы
в процессе генерации выполняли какие-то другие действия (печатать
на клавиатуре, движения мыши, обращения к дискам); это даст генератору
случайных чисел больше возможностей получить достаточное количество энтропии.
Необходимо получить много случайных чисел. Желательно, чтобы Вы
в процессе генерации выполняли какие-то другие действия (печатать
на клавиатуре, движения мыши, обращения к дискам); это даст генератору
случайных чисел больше возможностей получить достаточное количество энтропии.
gpg: /home/igfilatov/.gnupg/trustdb.gpg: создана таблица доверия
gpg: создан каталог '/home/igfilatov/.gnupg/openpgp-revocs.d'
gpg: сертификат отзыва записан в '/home/igfilatov/.gnupg/openpgp-revocs.d/72A66B675D58E03BA637EFBDD803BE9E0F020676.i
ev'.
открытый и секретный ключи созданы и подписаны.

pub   rsa4096 2025-02-20 [SC]
       72A66B675D58E03BA637EFBDD803BE9E0F020676
uid    Илья Филатов <ila689213@gmail.com>
sub    rsa4096 2025-02-20 [E]

igfilatov@igfilatov: $
[0] 0:~$
```

Рис. 3.4: Создание PGP ключа

## 3.4 Добавление PGP ключа в GitHub

Копирую свой PGP ключ используя инструмент xclip используется для копирования содержимого файла в буфер обмена, а флаг -selection clipboard делает так, чтобы скопированные данные были доступны для вставки через Ctrl + V. Добавляю данный ключ в github (рис. 3.5).

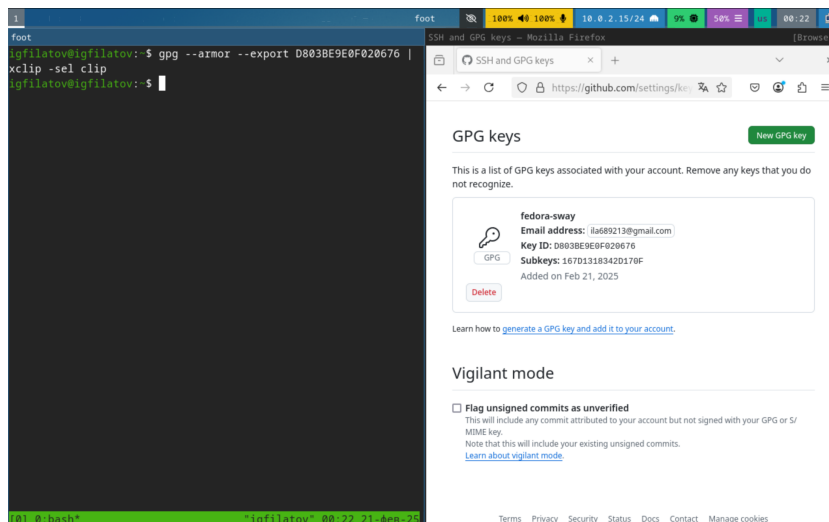
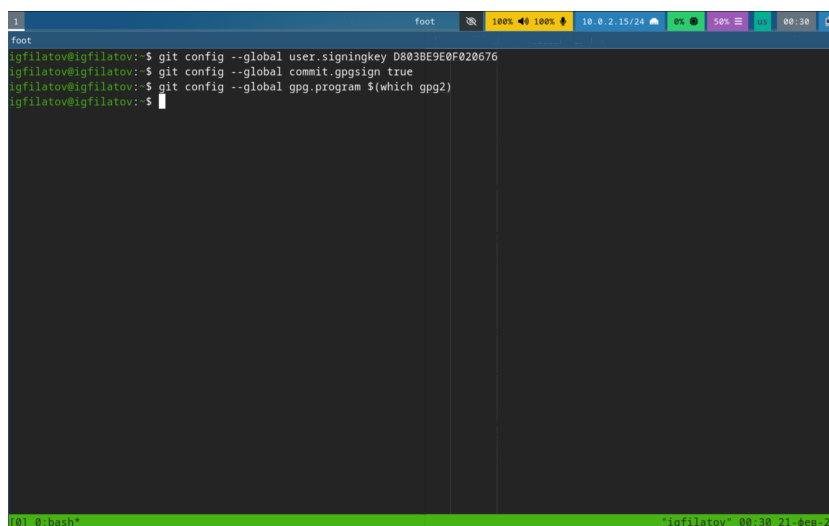


Рис. 3.5: Настройка PGP ключа в github



## 3.5 Настройка автоматических подписей коммитов git

Далее добавляю в конфиг авторизацию PGP (рис. 3.6).



```
foot
igfilatov@igfilatov:~$ git config --global user.signingkey D803BE9E0F020676
igfilatov@igfilatov:~$ git config --global commit.gpgsign true
igfilatov@igfilatov:~$ git config --global gpg.program $(which gpg2)
igfilatov@igfilatov:~$
```

Рис. 3.6: PGP авторизация

## 3.6 Настройка gh

Проходим авторизацию постепенно отвечая на вопросы утилиты (рис. 3.7).

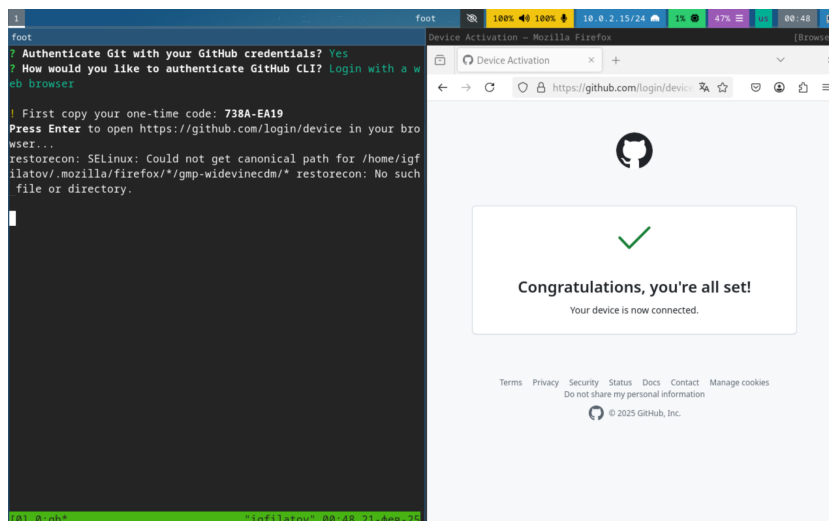


Рис. 3.7: Авторизация gh

## 3.7 Сознание репозитория курса на основе шаблона

После настройки создам свой репозиторий на основе шаблона (рис. 3.8).

```
1 foot
Клонирование в «os-intro»...
remote: Enumerating objects: 36, done.
remote: Counting objects: 100% (36/36), done.
remote: Compressing objects: 100% (35/35), done.
remote: Total 36 (delta 1), reused 21 (delta 0), pack-reused 0 (from 0)
Получение объектов: 100% (36/36), 19.37 Киб | 4.84 Миб/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/igfilatov/work/study/2024-2025/Операционные системы/os-intro/template/presentation»...
remote: Enumerating objects: 111, done.
remote: Counting objects: 100% (111/111), done.
remote: Compressing objects: 100% (77/77), done.
remote: Total 111 (delta 42), reused 100 (delta 31), pack-reused 0 (from 0)
Получение объектов: 100% (111/111), 102.17 Киб | 1.01 Миб/с, готово.
Определение изменений: 100% (42/42), готово.
Клонирование в «/home/igfilatov/work/study/2024-2025/Операционные системы/os-intro/template/report»...
remote: Enumerating objects: 142, done.
remote: Counting objects: 100% (142/142), done.
remote: Compressing objects: 100% (97/97), done.
remote: Total 142 (delta 60), reused 121 (delta 39), pack-reused 0 (from 0)
Получение объектов: 100% (142/142), 341.09 Киб | 2.07 Миб/с, готово.
Определение изменений: 100% (60/60), готово.
Submodule path 'template/presentation': checked out 'c9b2712b4b2d431ad5086c9c72a82bd2fcald4a6'
Submodule path 'template/report': checked out 'c26e22effe7b3e0495707d82ef561ab185f5c748'
igfilatov@igfilatov:~/work/study/2024-2025/Операционные системы$
[0] 0:bash*
```

Рис. 3.8: Создание собственного репозитория

Наконец настроим каталог курса перейдя в него и удалив лишние файлы, после создадим каталоги и отправим файлы на сервер (рис. 3.9).

```
1 foot
create mode 100644 project-personal/stage6/presentation/ texlabroot
create mode 100644 project-personal/stage6/presentation/Makefile
create mode 100644 project-personal/stage6/presentation/image/kulyabov.jpg
create mode 100644 project-personal/stage6/presentation/presentation.md
create mode 100644 project-personal/stage6/report/Makefile
create mode 100644 project-personal/stage6/report/bib/cite.bib
create mode 100644 project-personal/stage6/report/image/placeimg_800_600_tech.jpg
create mode 100644 project-personal/stage6/report/pandoc/csl/gost-r-7-0-5-2008-numeric.csl
create mode 100755 project-personal/stage6/report/pandoc/filters/pandoc_eqnos.py
create mode 100755 project-personal/stage6/report/pandoc/filters/pandoc_fignos.py
create mode 100755 project-personal/stage6/report/pandoc/filters/pandoc_secnos.py
create mode 100755 project-personal/stage6/report/pandoc/filters/pandoc_tablenos.py
create mode 100644 project-personal/stage6/report/pandoc/filters/pandocxnos/__init__.py
create mode 100644 project-personal/stage6/report/pandoc/filters/pandocxnos/core.py
create mode 100644 project-personal/stage6/report/pandoc/filters/pandocxnos/main.py
create mode 100644 project-personal/stage6/report/pandoc/filters/pandocxnos/pandocattributes.py
create mode 100644 project-personal/stage6/report/report.md
igfilatov@igfilatov:~/work/study/2024-2025/Операционные системы/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), 342.34 Киб | 2.98 Миб/с, готово.
Total 38 (delta 4), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (4/4), completed with 1 local object.
To github.com:igfilatov/study_2024-2025_os-intro.git
8e8ffa6..8a05c00 master -> master
igfilatov@igfilatov:~/work/study/2024-2025/Операционные системы/os-intro$
[0] 0:bash*
```

Рис. 3.9: Настройка каталога курса

## 3.8 Ответы на контрольные вопросы

1. Системы контроля версий (VCS) представляют собой программные инструменты, которые помогают отслеживать изменения в коде или других файлах проекта во времени, обеспечивая возможность совместной работы над проектом и восстановления предыдущих версий.
  - В системах контроля версий хранилище содержит все версии файлов, `commit` фиксирует конкретное состояние изменений, история представляет собой запись всех внесенных изменений, а рабочая копия является локальной версией файлов для непосредственной работы.
  - Централизованные VCS (например, SVN) используют один центральный сервер-хранилище, тогда как децентрализованные системы (например, Git) позволяют каждому разработчику иметь полную копию репозитория на своем компьютере.
  - При единоличной работе с хранилищем VCS пользователь последовательно создает изменения в рабочей копии, фиксирует их через `commit` и может при необходимости возвращаться к предыдущим версиям.
  - При работе с общим хранилищем VCS разработчики синхронизируют свои локальные изменения с центральным репозиторием через операции `pull` и `push`, что позволяет избежать конфликтов при параллельной работе над проектом.
  - Git решает основные задачи отслеживания изменений в коде, управления различными версиями проекта и организации эффективного процесса совместной разработки через систему веток и коммитов.
  - Команды `git` включают базовые операции создания и управления репозиторием (`init`, `clone`), регистрации изменений (`add`), фиксации состояний (`commit`), синхронизации с удаленным репозиторием (`push`, `pull`) и переключения между версиями (`checkout`).
  - При работе с локальным репозиторием используются команды `add` и `commit`

для фиксации изменений, а при взаимодействии с удаленным репозиторием добавляются команды `push` для отправки изменений и `pull` для получения обновлений из центрального хранилища.

- Ветви (`branches`) позволяют параллельно разрабатывать различные функциональности проекта независимо друг от друга, что особенно полезно при реализации новых возможностей или исправлении ошибок без влияния на стабильную версию кода.
- Игнорирование определенных файлов через `.gitignore` необходимо для исключения из системы контроля версий временных файлов, конфигураций IDE и других данных, которые не должны попадать в общий репозиторий проекта.

## 4 Выводы

Я приобрёл навыки установки и настройки применения и настройки средств контроля версий

## **5 Список литературы**

1. Архитектура ЭВМ