

Лабораторная работа №2

Бондарь Т. В.

Российский университет дружбы народов, Москва, Россия

Информация

- Бондарь Татьяна Владимировна
- НКАбд-01-24, студ. билет №1132246711
- Российский университет дружбы народов
- https://github.com/tvbondar/study_2024-2025_os-intro

.. ..

Выполнение лабораторной работы №1

Изучить идеологию и применения средств контроля версий, освоить умения по работе с git.

1. Создать базовую конфигурацию для работы с git.
2. Зарегистрироваться на GitHub.
3. Создать ключ SSH.
4. Создать ключ PGP.
5. Настроить подписи git.
6. Создать локальный каталог для выполнения заданий по предмету.

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

Выполнение лабораторной работы. Установка программного обеспечения

1. Устанавливаем git, перейдя на роль суперпользователя.

```
[root@vbox ~]# dnf install git
Обновление и загрузка репозитория:
Fedora 41 - x86_64 - Updates
777% [ <= > ] | 0.0 B/s | 0.0 B | 80m01s
```

Рис. 1: Установка git

2. Устанавливаем gh

```
Ничего делать.
[root@vbox ~]# dnf install gh
Обновление и загрузка репозитория:
Репозитории загружены.
Пакет Арх. Версия Репозиторий Размер
Установка:
gh x86_64 2.65.0-1.fc41 updates 42.6 MB
Сводка транзакции:
Установка: 1 пакета
```

Рис. 2: Установка gh

3. Задаем имя и email владельца репозитория. Настраиваем utf-8 в выводе сообщений гит. Задаем имя начальной ветки, параметры autocrlf, safecrlf.

```
[1/3] Проверить файлы пакета
[2/3] Подготовить транзакцию
[3/3] Установка gh-0:2.65.0-1.fc41.x86_64
Завершено!
[root@vbox ~]# git config --global user.name "Bondar Tatiana"
[root@vbox ~]# git config --global user.email "sinopahfox@yandex.ru"
[root@vbox ~]# git config --global core.quotepath false
[root@vbox ~]# git config --global init.defaultBranch master
[root@vbox ~]# git config --global core.autocrlf input
[root@vbox ~]# git config --global core.safecrlf warn
```

Рис. 3: Базовая настройка гит

4. По алгоритму rsa с ключем размером 4096 бит и по алгоритму ed25519

```
[root@vbox ~]# ssh-keygen -t rsa -b 4096
Generating public/private rsa key pair.
Enter file in which to save the key (/root/.ssh/id_rsa): /root/.ssh/id_rsa
Enter passphrase for "/root/.ssh/id_rsa" (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /root/.ssh/id_rsa
Your public key has been saved in /root/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:cnZDhFlE1EnI4hjK3UZ1TsiSVw5qTSnMS+k1FmRbtLQ root@vbox
The key's randomart image is:
+---[RSA 4096]-----+
|          *BBB      |
|      o  +000X  o    |
|      o  =+BB+.E    |
|      *  +o+  .     |
|      .  = S  o     |
|      +  +         |
|      o  +  .       |
|      .  .  .       |
|                      |
+-----[SHA256]-----+
[root@vbox ~]# ssh-keygen -t ed25519
unknown key type -ed25519
[root@vbox ~]# ssh-keygen -t ed25519
Generating public/private ed25519 key pair.
Enter file in which to save the key (/root/.ssh/id_ed25519): /root/.ssh/id_ed25519
```

5. Сгенерируем ключ

```
[root@vbox ~]# gpg --full-generate-key
gpg (GnuPG) 2.4.5; Copyright (C) 2024 g10 Code GmbH
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 '/root/.gnupg'
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
```

Рис. 5: Генерирование PGP-ключа

6. Учетная запись на Гитхабе и созданный заранее репозиторий курса

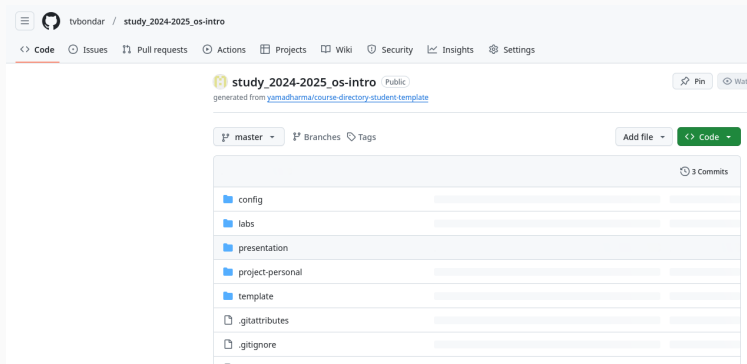



Рис. 6: Репозиторий курса

Добавление PGP ключа на GitHub

7. Вставляем сгенерированные GPG и SSH-ключи в поле ввода на Гитхабе, сохраняем их.


Authentication keys



New key
SHA256: 7eYQq6UFh6BqmNAjb5rpmpt9NurKBK5dbHsnLHwyua0
Added on Oct 8, 2024
Last used within the last 3 months — Read/write

SSH


Delete



key
SHA256: 6QJpxl31SL56T9ThmNFS70B15qvj8ae1L2NXUbrmYjE
Added on Nov 18, 2024
Last used within the last week — Read/write

SSH

Delete



sway spin
SHA256: d+Td4atLMnXS7efp9H+/7mTBxehYQeN10LYBgwc1bys
Added on Feb 17, 2025
Never used — Read/write

SSH

Delete

GPG keys

[New GPG key](#)

This is a list of GPG keys associated with your account. Remove any keys that you do not recognize.



GPG

ASDFGHJKL

Email address: `sinopahfox@yandex.ru`

Key ID: EF66AF49608ED501

[Delete](#)

Subkeys: 335386D6F10D30F4

Added on Feb 16, 2025

Learn how to [generate a GPG key and add it to your account](#).

Рис. 8: GPG-КЛЮЧ

8. Используя введенный email, укажем Git применять его при подписи коммитов

```
[root@vbox ~]# git config --global user.signingkey 33538BD6F10030F4
[root@vbox ~]# git config --global commit.gpgsign true
[root@vbox ~]# git config --global gpg.program $(which gpg2)
[root@vbox ~]# пр фер дпнат
-bash: пр: команда не найдена
[root@vbox ~]#
```

Рис. 9: Настройка автоматических подписей коммитов

9. Авторизовалась через браузер

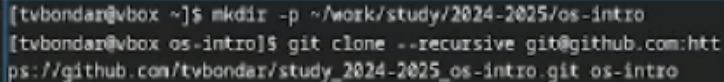
```
[tvbondar@vbox ~]$ gh auth login
? Where do you use GitHub? GitHub.com
? What is your preferred protocol for Git operations on this host? HTTPS
? Authenticate Git with your GitHub credentials? Yes
? How would you like to authenticate GitHub CLI? Login with a web browser

! First copy your one-time code: 1D66-A4B3
Press Enter to open https://github.com/login/device in your browser...
restorecon: SELinux: Could not get canonical path for /home/tvbondar/.mozilla/firefo
x/*/.gnupg-widewinecdm/* restorecon: No such file or directory.
```

Рис. 10: Авторизация

Шаблон для рабочего пространства. Создание репозитория курса на основе шаблона. Настройка каталога курса

10. Репозиторий курса уже был создан на Гитхаб. Необходимые файлы уже удалены. Создаю каталог, где будет храниться содержимое курса на моем устройстве.

A terminal window with a dark background and light-colored text. It shows two commands being executed. The first command creates a directory named 'os-intro' under the path '~/.work/study/2024-2025/'. The second command clones a repository from GitHub into the 'os-intro' directory.

```
[tvbondar@vbox ~]$ mkdir -p ~/.work/study/2024-2025/os-intro  
[tvbondar@vbox os-intro]$ git clone --recursive git@github.com:https://github.com/tvbondar/study_2024-2025_os-intro.git os-intro
```

Рис. 11: Создание каталога

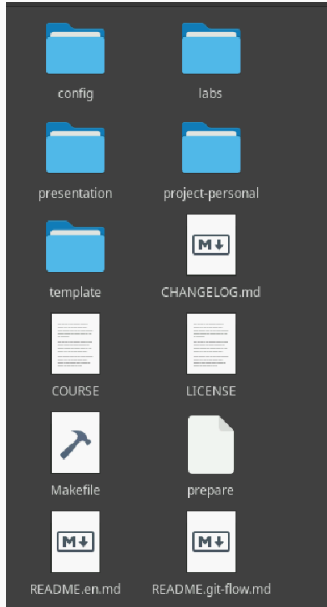


Рис. 12: Каталог курса на моем устройстве

1. Что такое системы контроля версий (VCS) и для решения каких задач они предназначены?
 - Система контроля версий — программное обеспечение для облегчения работы с изменяющейся информацией. Система управления версиями позволяет хранить несколько версий одного и того же документа, при необходимости возвращаться к более ранним версиям, определять, кто и когда сделал то или иное изменение, и многое другое. Системы контроля версий (Version Control System, VCS) применяются для:
 - Хранение полной истории изменений
 - причин всех производимых изменений
 - Откат изменений, если что-то пошло не так
 - Поиск причины и ответственного за появления ошибок в программе
 - Совместная работа группы над одним проектом
 - Возможность изменять код, не мешая работе других пользователей

2. Объясните следующие понятия VCS и их отношения: хранилище, commit, история, рабочая копия

- Репозиторий - хранилище версий - в нем хранятся все документы вместе с историей их изменения и другой служебной информацией.
- Commit — отслеживание изменений
- Рабочая копия - копия проекта, связанная с репозиторием (текущее состояние файлов проекта, основанное на версии из хранилища (обычно на последней))
- История хранит все изменения в проекте и позволяет при необходимости обратиться к нужным данным.

3. Что представляют собой и чем отличаются централизованные и децентрализованные VCS? Приведите примеры VCS каждого вида

- Централизованные VCS (Subversion; CVS; TFS; VAULT; AccuRev):
 - Одно основное хранилище всего проекта
 - Каждый пользователь копирует себе необходимые ему файлы из этого репозитория, изменяет и, затем, добавляет свои изменения обратно
- Децентрализованные VCS (Git; Mercurial; Bazaar):
- У каждого пользователя свой вариант (возможно не один) репозитория
 - Присутствует возможность добавлять и забирать изменения из любого репозитория . В классических системах контроля версий используется централизованная модель, предполагающая наличие единого репозитория для хранения файлов. Выполнение большинства функций по управлению версиями осуществляется специальным сервером. В отличие от классических, в распределённых системах контроля версий центральный репозиторий не является обязательным.

4. Опишите действия с VCS при единоличной работе с хранилищем.
 - Сначала создаем и подключаем удаленный репозиторий. Затем по мере изменения проекта отправлять эти изменения на сервер.
5. Опишите порядок работы с общим хранилищем VCS.
 - 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.
6. Каковы основные задачи, решаемые инструментальным средством git?
 - Первая — хранить информацию о всех изменениях в вашем коде, начиная с самой первой строчки, а вторая — обеспечение удобства командной работы над кодом.

7. Назовите и дайте краткую характеристику командам 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 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. Приведите примеры использования при работе с локальным и удалённым репозиториями.

- `git push -all` (push origin master/любой branch)

9. Что такое и зачем могут быть нужны ветви (branches)?

- Ветвление («ветка», branch) — один из параллельных участков истории в одном хранилище, исходящих из одной версии (точки ветвления). [3] • Обычно есть главная ветка (master), или ствол (trunk). • Между ветками, то есть их концами, возможно слияние. Используются для разработки новых функций.

10. Как и зачем можно игнорировать некоторые файлы при commit?

- 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

1. Кулябов Д. С. Введение в операционную систему UNIX - Лекция.
2. Таненбаум Э., Бос Х. Современные операционные системы. - 4-е изд. -СПб. : Питер, 2015. - 1120 с.

В ходе данной лабораторной работы мы изучили идеологию и применения средств контроля версий, освоили умения по работе с git.