отчёт по лабораторной работе n2

управления версиями

Джахангиров илгар Залид ”

Содержание

Список иллюстраций

1 Цель работы

- Изучить идеологию и применение средств контроля версий.
- Освоить умения по работе с git.
- Система контроля версий Git представляет собой набор программ командной строки.
- Благодаря тому, что Git является распределённой системой контроля версий, резер

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 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

Установка git

Установим git:

```
dnf install git
```

Установка gh

Fedora:

```
dnf install gh
```

Базовая настройка git

Зададим имя и email владельца репозитория:

```
git config --global user.name "Name Surname"
git config --global user.email "work@mail"
```

Настроим utf-8 в выводе сообщений git:

```
git config --global core.quotepath false
```

Настройте верификацию и подписание коммитов git (см. Верификация коммитов git с г

Зададим 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
```

Создайте ключи ssh

по алгоритму rsa с ключём размером 4096 бит:

```
ssh-keygen -t rsa -b 4096
```

по алгоритму ed25519:

```
ssh-keygen -t ed25519
```

Создайте ключи pgp

Генерируем ключ

```
gpg --full-generate-key
```

Из предложенных опций выбираем:

- тип RSA and RSA;
- размер 4096;

выберите срок действия; значение по умолчанию — 0 (срок действия не истекает

GPG запросит личную информацию, которая сохранится в ключе:

Имя (не менее 5 символов).

Адрес электронной почты.

При вводе email убедитесь, что он соответствует адресу, используемому на

Комментарий. Можно ввести что угодно или нажать клавишу ввода, чтобы оставить

Настройка github

Создайте учётную запись на https://github.com.

Заполните основные данные на https://github.com.

Добавление PGP ключа в GitHub

Выводим список ключей и копируем отпечаток приватного ключа:

```
gpg --list-secret-keys --keyid-format LONG
```

Отпечаток ключа — это последовательность байтов, используемая для идентификации б

Формат строки:

```
sec   Алгоритм/Отпечаток_ключа Дата_создания [Флаги] [Годен_до]
      ID_ключа
```

Скопируйте ваш сгенерированный PGP ключ в буфер обмена:

```
gpg --armor --export <PGP Fingerprint> | xclip -sel clip
```

Перейдите в настройки GitHub (https://github.com/settings/keys), нажмите на кнопк

Настройка автоматических подписей коммитов git

```
Используя введёный email, укажите Git применять его при подписи коммитов:

git config --global user.signingkey <PGP Fingerprint>
git config --global commit.gpgsign true
git config --global gpg.program $(which gpg2)
```

Настройка gh

```
Для начала необходимо авторизоваться

gh auth login

Утилита задаст несколько наводящих вопросов.
Авторизоваться можно через броузер.
```

/home/izdzhakhangirov/Загрузки/photo1676739178(1).jpeg /home/izdzhakhangirov/Загрузки/photo1676739178.jpeg /home/izdzhakhangirov/Загрузки/photo1676739179(7).jpeg /home/izdzhakhangirov/Загрузки/photo1676739179(6).jpeg /home/izdzhakhangirov/Загрузки/photo1676739179(5).jpeg /home/izdzhakhangirov/Загрузки/photo1676739179(4).jpeg /home/izdzhakhangirov/Загрузки/photo1676739179(3).jpeg /home/izdzhakhangirov/Загрузки/photo1676739179(2).jpeg /home/izdzhakhangirov/Загрузки/photo1676739191(5).jpeg /home/izdzhakhangirov/Загрузки/photo1676739191(4).jpeg /home/izdzhakhangirov/Загрузки/photo1676739191(3).jpeg /home/izdzhakhangirov/Загрузки/photo1676739191(2).jpeg /home/izdzhakhangirov/Загрузки/photo1676739191(1).jpeg /home/izdzhakhangirov/Загрузки/photo1676739191.jpeg /home/izdzhakhangirov/Загрузки/photo1676739192(2).jpeg /home/izdzhakhangirov/Загрузки/photo1676739192(1).jpeg /home/izdzhakhangirov/Загрузки/photo1676739539(1).jpeg /home/izdzhakhangirov/Загрузки/photo1676739179(1).jpeg /home/izdzhakhangirov/Загрузки/photo1676739179.jpeg /home/izdzhakhangirov/Загрузки/photo1676739192.jpeg /home/izdzhakhangirov/Загрузки/photo1676739539(2).jpeg
Выводы/

Я создал учотную запись устоновил програму .с генировал и устоновил програм-ный обиспечение дальнешим работать git hub # Список литературы{.unnumbered}