Отчёт по лабораторной работе 2

дисциплина: Операционные системы

Волгин Андрей НПИбд-01-22

Содержание

1 Цель работы:

Целью данной работы является изучение идеологии и применения средств контроля версий и освоение умений по работе с git.

2 Порядок выполнения лабораторной работы:

Заходим под правами суперпользователя, затем устанавливаем git и gh (рис. 1-2)

```
[aavolgin@aavolgin ~]$ sudo -i
[sudo] пароль для aavolgin:
[root@aavolgin ~]# dnf install git
Fedora 37 - x86_64 - Updates                            19 kB/s |  14 kB     00:00
Fedora 37 - x86_64 - Updates                           3.1 MB/s | 4.8 MB     00:01
Fedora Modular 37 - x86_64 - Updates                    62 kB/s |  17 kB     00:00
Fedora Modular 37 - x86_64 - Updates                   102 kB/s |  99 kB     00:00
Последняя проверка окончания срока действия метаданных: 0:00:01 назад, Сб 25 фев 2023 16:38:4
5.
Пакет git-2.39.2-1.fc37.x86_64 уже установлен.
Зависимости разрешены.
Отсутствуют действия для выполнения.
Выполнено!
[root@aavolgin ~]# dnf install gh
Последняя проверка окончания срока действия метаданных: 0:06:29 назад, Сб 25 фев 2023 17:13:3
0.
Зависимости разрешены.
================================================================================
 Пакет           Архитектура         Версия                    Репозиторий          Размер
================================================================================
Установка:
 gh              x86_64              2.23.0-1.fc37             updates              8.3 M

Результат транзакции
================================================================================
Установка  1 Пакет
```

Затем зададим имя и email владельца репозитория, настроим utf-8 в выводе сообщений git, зададим имя начальной ветки (будем называть её master) и установим параметры autocrlf и safecrlf (рис. 3)

```
[root@aavolgin ~]# git config --global user.name "Андрей Волгин"
[root@aavolgin ~]# git config --global user.email "kickasscococo@mail.ru"
[root@aavolgin ~]# git config --global core.quotepath false
[root@aavolgin ~]# git config --global init.defaultBranch master
[root@aavolgin ~]# git config --global core.autocrlf input
[root@aavolgin ~]# git config --global core.safecrlf warn
```

Рис. 3. Базовая настройка git

Создадим ключ ssh и ключ pgp (рис. 4)

```
[root@aavolgin ~]# ssh-keygen -t rsa -b 4096
Generating public/private rsa key pair.
Enter file in which to save the key (/root/.ssh/id_rsa): /home/aavolgin/.ssh/id_rsa
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Saving key "/home/aavolgin/.ssh/id_rsa" failed: No such file or directory
[root@aavolgin ~]# 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
Created directory '/root/.ssh'.
Enter passphrase (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:
SHA256:6Mkw+woDGkv1Fan6nrO536WCLCwYw7HqNE4wajPtEbo root@aavolgin.aavolgin.net
The key's randomart image is:
+---[RSA 4096]----+
|        ..       |
|        ..       |
|    .  ..        |
| .. .... .       |
|*.o.+.. S        |
|BBo..* .         |
|*@+o+ =    .     |
|B.*=.*o.. o      |
|.E..+B*..o       |
+----[SHA256]-----+
[root@aavolgin ~]#
```

```
[root@aavolgin ~]# gpg --full-generate-key
gpg (GnuPG) 2.3.8; Copyright (C) 2021 Free Software Foundation, Inc.
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.

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
Запрошенный размер ключа - 4096 бит
Выберите срок действия ключа.
         0 = не ограничен
      <n>  = срок действия ключа - n дней
      <n>w = срок действия ключа - n недель
      <n>m = срок действия ключа - n месяцев
      <n>y = срок действия ключа - n лет
Срок действия ключа? (0) 0
Срок действия ключа не ограничен
Все верно? (y/N) y
```

При создании ключа gpg нужно будет составить идентификатор пользователя (рис. 6)

```
GnuPG должен составить идентификатор пользователя для идентификации ключа.

Ваше полное имя: Андрей Волгин
Адрес электронной почты: kickasscococo@mail.ru
Примечание:
Используется таблица символов 'utf-8'.
Вы выбрали следующий идентификатор пользователя:
    "Андрей Волгин <kickasscococo@mail.ru>"

Сменить (N)Имя, (C)Примечание, (E)Адрес; (O)Принять/(Q)Выход? o
Необходимо получить много случайных чисел. Желательно, чтобы Вы
в процессе генерации выполняли какие-то другие действия (печать
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 (печать
на клавиатуре, движения мыши, обращения к дискам); это даст генератору
случайных чисел больше возможностей получить достаточное количество энтропии.
gpg: /root/.gnupg/trustdb.gpg: создана таблица доверия
gpg: создан каталог '/root/.gnupg/openpgp-revocs.d'
gpg: сертификат отзыва записан в '/root/.gnupg/openpgp-revocs.d/0D5E83505F176618875DF3B183472
402DCD69AB3.rev'.
открытый и секретный ключи созданы и подписаны.

pub   rsa4096 2023-02-25 [SC]
      0D5E83505F176618875DF3B183472402DCD69AB3
uid                      Андрей Волгин <kickasscococo@mail.ru>
sub   rsa4096 2023-02-25 [E]
```

Рис. 6. Составление идентификатора пользователя

Выводим список ключей и копируем отпечаток приватного ключа (рис. 7)

```
[root@aavolgin ~]# gpg --list-secret-keys --keyid-format LONG
gpg: проверка таблицы доверия
gpg: marginals needed: 3  completes needed: 1  trust model: pgp
gpg: глубина: 0  достоверных:   1  подписанных:   0  доверие: 0-, 0q, 0n, 0m, 0f, 1u
/root/.gnupg/pubring.kbx
------------------------
sec   rsa4096/83472402DCD69AB3 2023-02-25 [SC]
      0D5E83505F176618875DF3B183472402DCD69AB3
uid                 [  абсолютно ] Андрей Волгин <kickasscococo@mail.ru>
ssb   rsa4096/E27D5F97D8E6B40D 2023-02-25 [E]

[root@aavolgin ~]#
```

Рис. 7. Список ключей

И затем скопируем gpg ключ в свой аккаунт на гитхабе (рис. 8)

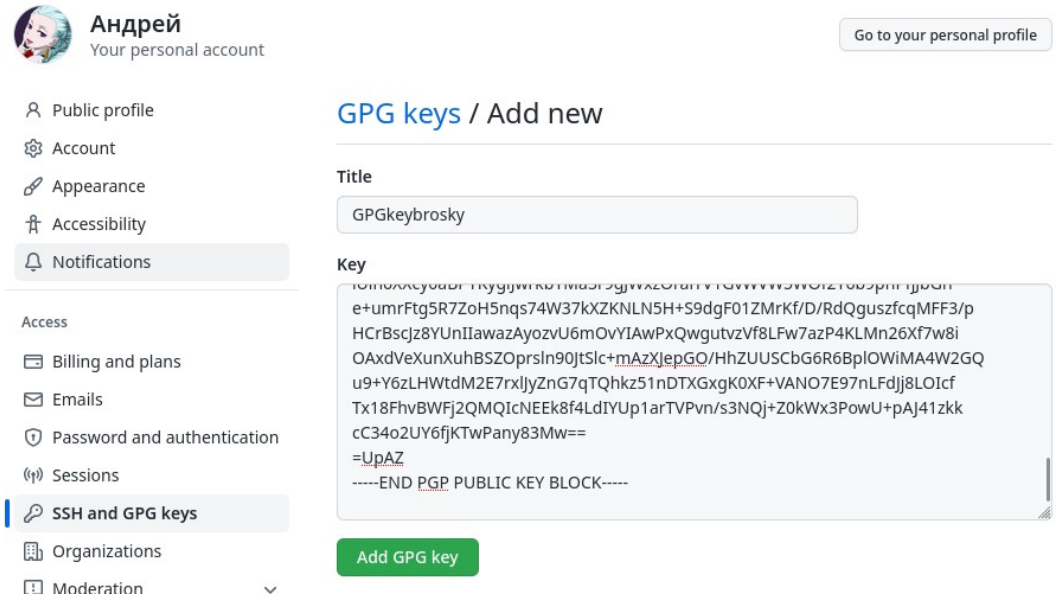

Рис. 8. Копирование ключа pgp

Настроим автоматическую подпись коммитов (рис. 9)

```
[root@aavolgin ~]# git config --global user.signingkey 83472402DCD69AB3
[root@aavolgin ~]# git config --global commit.gpgsign true
[root@aavolgin ~]# git config --global gpg.program $(which gpg2)
[root@aavolgin ~]#
```

Рис. 9. Настройка подписей коммитов

Проведем авторизацию (рис. 10)

```
Выполнено!
[root@aavolgin ~]# gh auth login
? What account do you want to log into? GitHub.com
? What is your preferred protocol for Git operations? 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: F1E9-BB39
Press Enter to open github.com in your browser...
```

Рис. 10. Создание ключа pgp

Затем создадим репозиторий курса на основе заданного шаблона (рис. 11)

```
[root@aavolgin Операционные системы]# git clone --recursive git@github.com:ShooterFromChicago
/study_2022-2023_os-intro.git os-intro
Клонирование в «os-intro»...
remote: Enumerating objects: 27, done.
remote: Counting objects: 100% (27/27), done.
remote: Compressing objects: 100% (26/26), done.
remote: Total 27 (delta 1), reused 11 (delta 0), pack-reused 0
Получение объектов: 100% (27/27), 16.96 КиБ | 8.48 МиБ/с, готово.
Определение изменений: 100% (1/1), готово.
Подмодуль «template/presentation» (https://github.com/yamadharma/academic-presentation-markdo
wn-template.git) зарегистрирован по пути «template/presentation»
Подмодуль «template/report» (https://github.com/yamadharma/academic-laboratory-report-templat
e.git) зарегистрирован по пути «template/report»
Клонирование в «/root/work/study/2022-2023/Операционные системы/os-intro/template/presentatio
n»...
remote: Enumerating objects: 82, done.
remote: Counting objects: 100% (82/82), done.
remote: Compressing objects: 100% (57/57), done.
remote: Total 82 (delta 28), reused 77 (delta 23), pack-reused 0
Получение объектов: 100% (82/82), 92.90 КиБ | 1.21 МиБ/с, готово.
Определение изменений: 100% (28/28), готово.
Клонирование в «/root/work/study/2022-2023/Операционные системы/os-intro/template/report»...
remote: Enumerating objects: 101, done.
remote: Counting objects: 100% (101/101), done.
remote: Compressing objects: 100% (70/70), done.
remote: Total 101 (delta 40), reused 88 (delta 27), pack-reused 0
Получение объектов: 100% (101/101), 327.25 КиБ | 2.11 МиБ/с, готово.
Определение изменений: 100% (40/40), готово.
Submodule path 'template/presentation': checked out 'b1be3800ee91f5809264cb755d316174540b753e
'
Submodule path 'template/report': checked out '1d1b61dcac9c287a83917b82e3aef11a33b1e3b2'
[root@aavolgin Операционные системы]#
```

Рис. 11. Создание ключа pgp

Затем удалим лишние файлы: rm package.json Создадим необходимые каталоги: echo os-intro > COURSE make И отправим файлы на сервер (рис 12-13): git add . git commit -am ‘feat(main): make course structure’ git push

```
[root@aavolgin os-intro]# git commit -am 'feat(main): make course structure'
[master bd716fc] feat(main): make course structure
 361 files changed, 100327 insertions(+), 14 deletions(-)
[root@aavolgin os-intro]# git push
Перечисление объектов: 40, готово.
Подсчет объектов: 100% (40/40), готово.
При сжатии изменений используется до 3 потоков
Сжатие объектов: 100% (30/30), готово.
Запись объектов: 100% (38/38), 343.62 КиБ | 2.44 МиБ/с, готово.
Всего 38 (изменений 4), повторно использовано 0 (изменений 0), повторно использовано пакетов
0
remote: Resolving deltas: 100% (4/4), completed with 1 local object.
To github.com:ShooterFromChicago/study_2022-2023_os-intro.git
   9604338..bd716fc  master -> master
[root@aavolgin os-intro]#
```

3 Ответы на контрольные вопросы:

1. Что такое системы контроля версий (VCS) и для решения каких задач они предназначаются? Это программное обеспечение для облегчения работы с изменяющейся информацией. VCS 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.

2. Объясните следующие понятия VCS и их отношения: хранилище, commit, история, рабочая копия. Хранилище (repository), или репозитарий, — место хранения всех версий и служебной информации.
   Commit («[трудовой] вклад», не переводится) — синоним версии; процесс создания новой версии. История – место, где сохраняются все коммиты, по которым можно посмотреть данные о коммитах. Рабочая копия – текущее состояние файлов проекта, основанное на версии, загруженной из хранилища.
3. Что представляют собой и чем отличаются централизованные и децентрализованные VCS? Приведите примеры VCS каждого вида.
   Централизованные VCS:
   одно основное хранилище всего проекта и каждый пользователь копир ует себе необходимые ему файлы из этого репозитория, изменяет и, затем, добавляет свои изменения обратно. Децентрализованные VCS: у каждого пользователя свой вариант (возможно не один) репозитория.
4. Опишите действия с VCS при единоличной работе с хранилищем.
5. Опишите порядок работы с общим хранилищем VCS.
6. Каковы основные задачи, решаемые инструментальным средством git?
   Git — это система управления версиями. 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. Назовите и дайте краткую характеристику командам git. git –version (Проверка версии Git) git init (Инициализировать ваш текущий рабочий каталог как Git-репозиторий) git
   clone https://www.github.com/username/repo-name (Скопировать существующий удаленный Git-репозиторий) git remote (Просмотреть список текущих удалённых репозиториев Git) git remote -v (Для более подробного вывода) git add my_script.py (Можете указать в команде конкретный файл). git add . (Позволяет охватить все файлы в текущем каталоге, включая файлы, чье имя начинается с точки) git commit -am “Commit message” (Вы можете сжать все индексированные файлы и отправить коммит). git branch (Просмотреть список текущих веток можно с помощью команды branch) git –help (Чтобы узнать больше обо всех доступных параметрах и командах) git push origin master (Передать локальные коммиты в ветку удаленного репозитория).
8. Приведите примеры использования при работе с локальным и удалённым репозиториями.
9. Что такое и зачем могут быть нужны ветви (branches)? Ветки нужны, чтобы несколько программистов могли вести работу над одним и тем же проектом или даже файлом одновременно, при этом не мешая друг другу. Кроме того, ветки используются для тестирования экспериментальных функций: чтобы не повредить основному проекту, создается новая ветка специально для экспериментов.

10. Как и зачем можно игнорировать некоторые файлы при commit? Игнорируемые файлы — это, как правило, артефакты сборки и файлы, генерируемые машиной из исходных файлов в вашем репозитории, либо файлы, которые по какой-либо иной причине не должны попадать в коммиты.

4 Вывод:

Во время выполнения лабораторной работы были изучены идеология и применение средств контроля версий, а также освоены умения по работе с git.