

Отчёт по лабораторной работе №3

Дисциплина: Операционные системы

Нечаева Кира Андреевна

Содержание

1	1 Цель работы	5
2	2 Задание	6
3	3 Теоретическое введение	7
4	4 Выполнение лабораторной работы	9
4.1	1. Установка git	9
4.2	2. Базовая настройка git	9
4.3	3. Создание SSH ключа	10
4.4	4. Создание GPG ключа	10
4.5	5. Настройка gh	12
4.6	6. Создание рабочего пространства на основе шаблона	12
4.7	7. Создание репозитория курса на основе шаблона	12
4.8	8. Настройка каталога курса	13
5	6 Вывод	14
	Список литературы	15

Список иллюстраций

Список таблиц

1 1 Цель работы

Целью работы является изучить идеологию и применение средств контроля версий. Приобрести практические навыки по работе с системой git.

2 2 Задание

1. Установка git
2. Базовая настройка git
3. Создание SSH ключа
4. Создание GPG ключа
5. Настройка git
6. Создание рабочего пространства на основе шаблона
7. Создание репозитория курса на основе шаблона
8. Настройка каталога курса

3 3 Теоретическое введение

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 зави-

симости 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 4 Выполнение лабораторной работы

4.1 1. Установка git

Поскольку у меня уже установлен Github, я могу пропустить этот шаг, однако я

```
[kanechaeva@192 ~]$ sudo dnf install git
[sudo] пароль для kanechaeva:
Fedora 38 - x86_64 - Updates                81 kB/s | 19 kB    00:00
Fedora 38 - x86_64 - Updates                1.9 MB/s | 4.3 MB  00:02
Fedora Modular 38 - x86_64 - Updates        30 kB/s | 20 kB    00:00
Пакет git-2.42.0-2.fc38.x86_64 уже установлен.
Пакет git-2.43.0-1.fc38.x86_64 уже установлен.
Зависимости разрешены.
Нет действий для выполнения.
Выполнено!
[kanechaeva@192 ~]$ sudo dnf install gh
Последняя проверка окончания срока действия метаданных: 0:00:14 назад, Пт 23 фев 2024 14:27:21.
Пакет gh-2.36.0-1.fc38.x86_64 уже установлен.
Зависимости разрешены.
Нет действий для выполнения.
Выполнено!
```

покажу, что Git у меня есть. (рис. [??])

4.2 2. Базовая настройка git

Сначала сделаю предварительную конфигурацию git. Открываю терминал и ввожу команды: `git config --global user.name " "` `git config --global user.email "kirusya1234@gmail.com"` указав свои имя и email. Настраиваю utf-8 в выводе сообщений git для верного отображения символов. Задаю имя начальной ветки "master". Задаю параметр `autocrlf` со значением `input`. Задаю параметр `safecrlf` со значением `warn`, так Git будет проверять преобразование на обратимость. Все действия

отображены в рис. [??]

```
[kanechaeva@192 ~]$ git config --global user.email "kirusya1234@gmail.com"
[kanechaeva@192 ~]$ git config --global core.quotepath false
[kanechaeva@192 ~]$ git config --global init.defaultBranch master
[kanechaeva@192 ~]$ git config --global core.autocrlf input
[kanechaeva@192 ~]$ git config --global core.safecrlf warn
```

4.3 3. Создание SSH ключа

Для последующей идентификации пользователя на сервере репозитория генерирую пару SSH ключей: по алгоритму rsa (рис. [??]) и по алгоритму

```
[kanechaeva@192 ~]$ ssh-keygen -t rsa -b 4096
Generating public/private rsa key pair.
Enter file in which to save the key (/home/kanechaeva/.ssh/id_rsa):
/home/kanechaeva/.ssh/id_rsa already exists.
Overwrite (y/n)? y
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/kanechaeva/.ssh/id_rsa
Your public key has been saved in /home/kanechaeva/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:Yu0HC1jQe1T5omf4psGENju+aaHs8kGMVtxt0D6Q0Ek kanechaeva@192.168.1.14
The key's randomart image is:
+---[RSA 4096]-----+
|  .+E+  ..          |
|  ...=oo.          |
|  .O..+o  .        |
|  o.. +.o.  .       |
|  ..o * *oS.        |
|  .. +.Oo+o         |
|  .O.+.=+          |
|  . oo.+ +o         |
|  +o.+oo           |
+---[SHA256]-----+
```

ed25519 (рис. [??]).

```
[kanechaeva@192 ~]$ ssh-keygen -t ed25519
Generating public/private ed25519 key pair.
Enter file in which to save the key (/home/kanechaeva/.ssh/id_ed25519):
/home/kanechaeva/.ssh/id_ed25519 already exists.
Overwrite (y/n)? y
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/kanechaeva/.ssh/id_ed25519
Your public key has been saved in /home/kanechaeva/.ssh/id_ed25519.pub
The key fingerprint is:
SHA256:0kCNI2+TZb85uFXJ91VW8zrJAKL1zn8X4pMoRypf1U4 kanechaeva@192.168.1.14
The key's randomart image is:
+--[ED25519 256]--+
|      . . .      |
|  o . . . . +   |
|  o . . . . +   |
|  o . . o = .+. |
|  . o o S +.=o=E |
|  . = o .o.++=o. |
|  = oooo + +o.o |
|  . ..o+o+  ... |
|  .oo         |
+---[SHA256]-----+
```

4.4 4. Создание GPG ключа

Затем генерирую gpg ключ. Из предложенных опций выбираю тип RSA and RSA, размер 4096 и безвременный срок действия. Затем ввожу некоторую личную информацию и придумываю пароль. (рис. [??])

```
[kanechaeva@192 ~]$ gpg --full-generate-key
gpg (GnuPG) 2.4.0; 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

GnuPG должен составить идентификатор пользователя для идентификации ключа.
Ваше полное имя: Kira
Адрес электронной почты: kirusya1234@gmail.com
Примечание:
Вы выбрали следующий идентификатор пользователя:
  "Kira <kirusya1234@gmail.com>"

Сменить (N)Имя, (C)Примечание, (E)Адрес; (O)Принять/(Q)Выход? O
```

Теперь я вывожу список ключей и копирую отпечаток приватного ключа в

```
[kanechaeva@192 ~]$ 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: 3 подписанных: 0 доверие: 0-, 0q, 0n, 0m, 0f, 3u
/home/kanechaeva/.gnupg/pubring.kbx
-----
sec   rsa4096/0DC6C3BBD1176A3D 2024-02-20 [SC]
      50C24C00DC947BFA68EACCA20DC6C3BBD1176A3D
uid   [ абсолютно ] Kira <1132236031@pfur.ru>
ssb   rsa4096/58126A00F705C7E1 2024-02-20 [E]

sec   rsa4096/2FC6A1C8CC59AA1B 2024-02-20 [SC]
      117C15ECBC22A791337B31C82FC6A1C8CC59AA1B
uid   [ абсолютно ] Kira <1132236031@pfur.ru>
ssb   rsa4096/8F318E8A653FD71B 2024-02-20 [E]

sec   rsa4096/9EB48FF5ABD28E56 2024-02-23 [SC]
      31E93F7AD4540121BCC312929EB48FF5ABD28E56
uid   [ абсолютно ] Kira <kirusya1234@gmail.com>
ssb   rsa4096/A59266FA1A5FD06D 2024-02-23 [E]
```


буфер обмена. (рис. [??])

Захожу на сайт GitHub. Выбираю в меню “Настройки”, затем “SSH and GPG keys”. Нажимаю кнопку «New GPG key». Вставляю скопированный ключ в поле «Key». Называю его key1. Нажимаю «Add GPG-key», чтобы завершить добавление ключа. К сожалению, я забыла сфотографировать процесс добавления ключа на Github, так что прикрепляю его уже в созданном виде. (рис. [??])

GPG keys

[New GPG key](#)

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



key1
Email address: kirusya1234@gmail.com
Key ID: 9EB48FF5ABD28E56
Subkeys: A59266FA1A5FD06D
Added on Feb 23, 2024

Delete

4.5 5. Настройка gh

Используя введённый email, указываю Git применять его при подписи коммитов. Теперь необходимо авторизоваться. Ввожу: `gh auth login`. Утилита задаст несколько наводящих вопросов, после чего я авторизовываюсь через браузер.

```
[kanechaeva@192 ~]$ gh auth login
? What account do you want to log into? GitHub.com
? You're already logged into github.com. Do you want to re-authenticate? Yes
? What is your preferred protocol for Git operations? HTTPS
? How would you like to authenticate GitHub CLI? Login with a web browser

! First copy your one-time code: 786E-0ECF
Press Enter to open github.com in your browser...
✓ Authentication complete.
- gh config set -h github.com git_protocol https
✓ Configured git protocol
✓ Logged in as kanechaeva
[kanechaeva@192 ~]$
```

(рис. [??])

4.6 6. Создание рабочего пространства на основе шаблона

Репозиторий на основе шаблона создаю через web-интерфейс github. Перехожу на станицу репозитория с шаблоном курса <https://github.com/yamadharma/course-directory-student-template>. Далее выбираю Use this template.

4.7 7. Сознание репозитория курса на основе шаблона

Теперь создаю рабочее пространство (рис. [??]). Я делаю это по типологии для 2022-2023 года. (рис. [??])

```
[kanechaeva@192 ~]$ mkdir -p ~/work/study/2023-2024/"Операционные системы"
[kanechaeva@192 ~]$ cd ~/work/study/2023-2024/"Операционные системы"
```

```
[kanechaeva@192 Операционные системы]$ gh repo create study_2023-2024_os-intro --template=yamadharma/course-directory-student-template --public
Created repository kanechaeva/study_2023-2024_os-intro on GitHub
```

4.8 8. Настройка каталога курса

Перехожу в каталог курса, удаляю лишние файлы (рис. [??]) и создаю необходимые каталоги (рис. [??]). После чего отправляю файлы на сервер (рис. [??]).

```
[kanechaeva@192 Операционные системы]$ cd ~/work/study/2023-2024/"Операционные системы"/study_2023-2024_os-intro
[kanechaeva@192 study_2023-2024_os-intro]$ rm package.json

[kanechaeva@192 study_2023-2024_os-intro]$ git add .
[kanechaeva@192 study_2023-2024_os-intro]$ git commit -am 'feat: make course structure'
[master 778213c] feat(main): make course structure
8 files changed, 7 insertions(+), 1 deletion(-)
create mode 100644 labs/README.md
create mode 100644 labs/README.ru.md
create mode 100644 prepare
create mode 100644 presentation/README.md
create mode 100644 presentation/README.ru.md
create mode 100644 project-personal/README.md
create mode 100644 project-personal/README.ru.md
[kanechaeva@192 study_2023-2024_os-intro]$ git push
Перечисление объектов: 100% (15/15), готово.
Подсчет объектов: 100% (15/15), готово.
При сжатии изменений используется до 12 потоков
Сжатие объектов: 100% (6/6), готово.
Запись объектов: 100% (13/13), 1.56 КиБ | 1.56 МБ/с, готово.
Всего 13 (изменений 1), повторно использовано 1 (изменений 0),
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To https://github.com/kanechaeva/study_2023-2024_os-intro.git
  9e6fda6..778213c master -> master

0b11c92..9e6fda6 master -> master
[kanechaeva@192 study_2023-2024_os-intro]$ echo os-intro > COURSE
[kanechaeva@192 study_2023-2024_os-intro]$ make prepare
```

5 6 Вывод

При выполнении данной лабораторной работы я изучила идеологию и применение средств контроля версий и приобрела практические навыки по работе с системой git.

Список литературы