

Цель работы

- Изучить применение средств контроля версий
- Освоить умения по работе с git'ом

Выполнение лабораторной работы

Настройка GIT

Сконфигурируем `git` и создадим `SSH` ключ

```
[root@irulev ~]# git config --global user.name "irulev"
[root@irulev ~]# git config --global user.email "rulev02@mail"
[root@irulev ~]# git config --global core.quotepath false
[root@irulev ~]# git config --global init.defaultBranch master
[root@irulev ~]# ssh-keygen -t rsa -b 4096
Generating public/private rsa key pair.
Enter file in which to save the key (/root/.ssh/id_rsa): lab_key
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in lab_key
Your public key has been saved in lab_key.pub
The key fingerprint is:
SHA256:MLkAjCCtdbZTNFxH+A+wZA77LU+C5A91q2QQTPOxEpo root@irulev
The key's randomart image is:
+---[RSA 4096]-----+
|++. o*+..oo      |
|o +ooo*o=.       |
| o.ooo*B +       |
|. .oo+*+ +       |
|  E+o=So +       |
|    + * + .       |
|    = *           |
|    o .           |
|                  |
+---[SHA256]-----+
```

Создадим PGP ключ

```

[root@irulev ~]# gpg --full-generate-key
gpg (GnuPG) 2.4.3; Copyright (C) 2023 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.

Please select what kind of key you want:
  (1) RSA and RSA
  (2) DSA and Elgamal
  (3) DSA (sign only)
  (4) RSA (sign only)
  (9) ECC (sign and encrypt) *default*
 (10) ECC (sign only)
 (14) Existing key from card
Your selection? 1
RSA keys may be between 1024 and 4096 bits long.
What keysize do you want? (3072) 4096
Requested keysize is 4096 bits
Please specify how long the key should be valid.
  0 = key does not expire
<n> = key expires in n days
<n>w = key expires in n weeks
<n>m = key expires in n months
<n>y = key expires in n years
Key is valid for? (0) 0
Key does not expire at all
Is this correct? (y/N) y

GnuPG needs to construct a user ID to identify your key.

Real name: Ivan
Email address: rulev02@mail.ru
Comment:
You selected this USER-ID:
  "Ivan <rulev02@mail.ru>"

Change (N)ame, (C)omment, (E)mail or (O)kay/(Q)uit?
Change (N)ame, (C)omment, (E)mail or (O)kay/(Q)uit? o
We need to generate a lot of random bytes. It is a good idea to perform
some other action (type on the keyboard, move the mouse, utilize the
disks) during the prime generation; this gives the random number
generator a better chance to gain enough entropy.
We need to generate a lot of random bytes. It is a good idea to perform
some other action (type on the keyboard, move the mouse, utilize the
disks) during the prime generation; this gives the random number
generator a better chance to gain enough entropy.
gpg: /root/.gnupg/trustdb.gpg: trustdb created
gpg: directory '/root/.gnupg/openpgp-revocs.d' created
gpg: revocation certificate stored as '/root/.gnupg/openpgp-revocs.d/902C632D32A2EEFA0C5DE64BC53165A207B61E0B.rev'
public and secret key created and signed.

pub
uid
sub


[root@irulev ~]#

```

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

скопируем PGP ключ.

И вставим в наш Github.



irulev (irulev)
 Your personal account

- Public profile
- Account
- Appearance
- Accessibility
- Notifications

Access

- Billing and plans
- Emails
- Password and authentication
- Sessions
- SSH and GPG keys**
- Organizations
- Enterprises
- Moderation

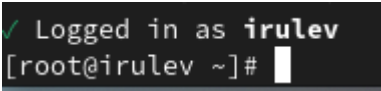
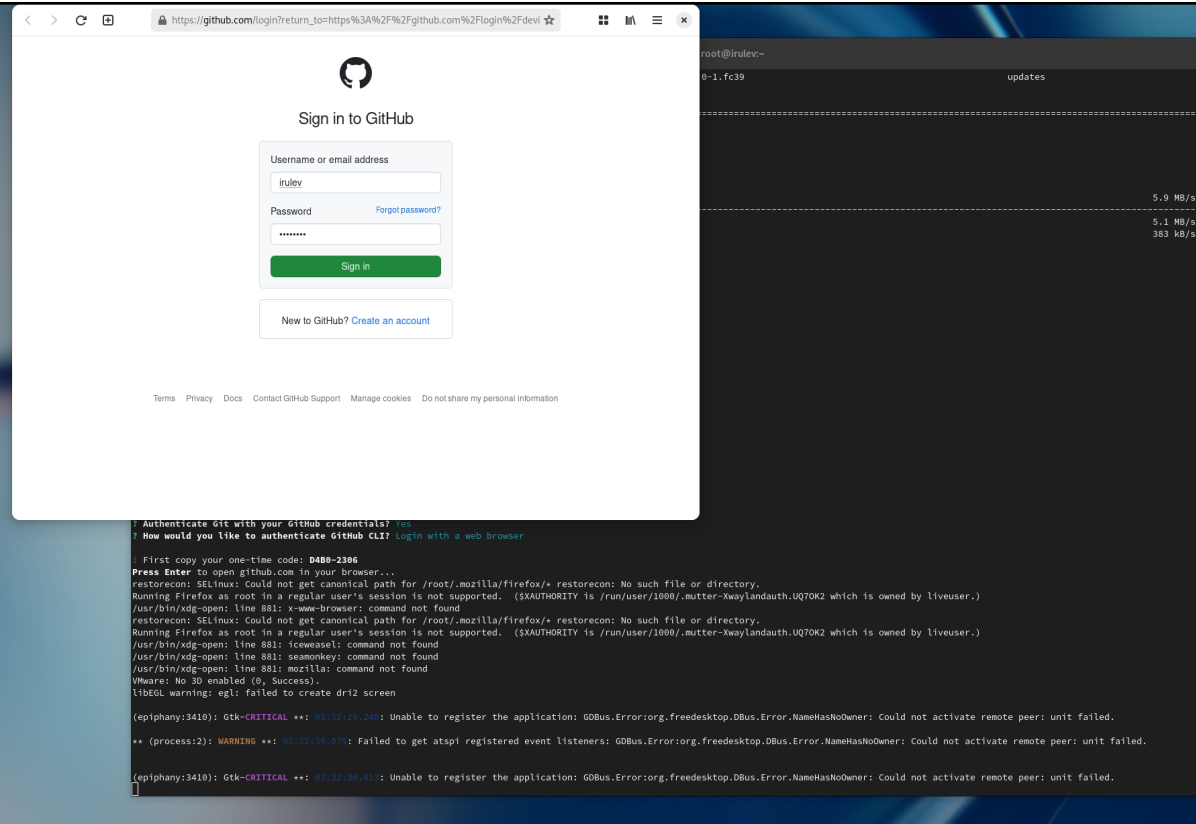
Add new GPG key

Title

Key

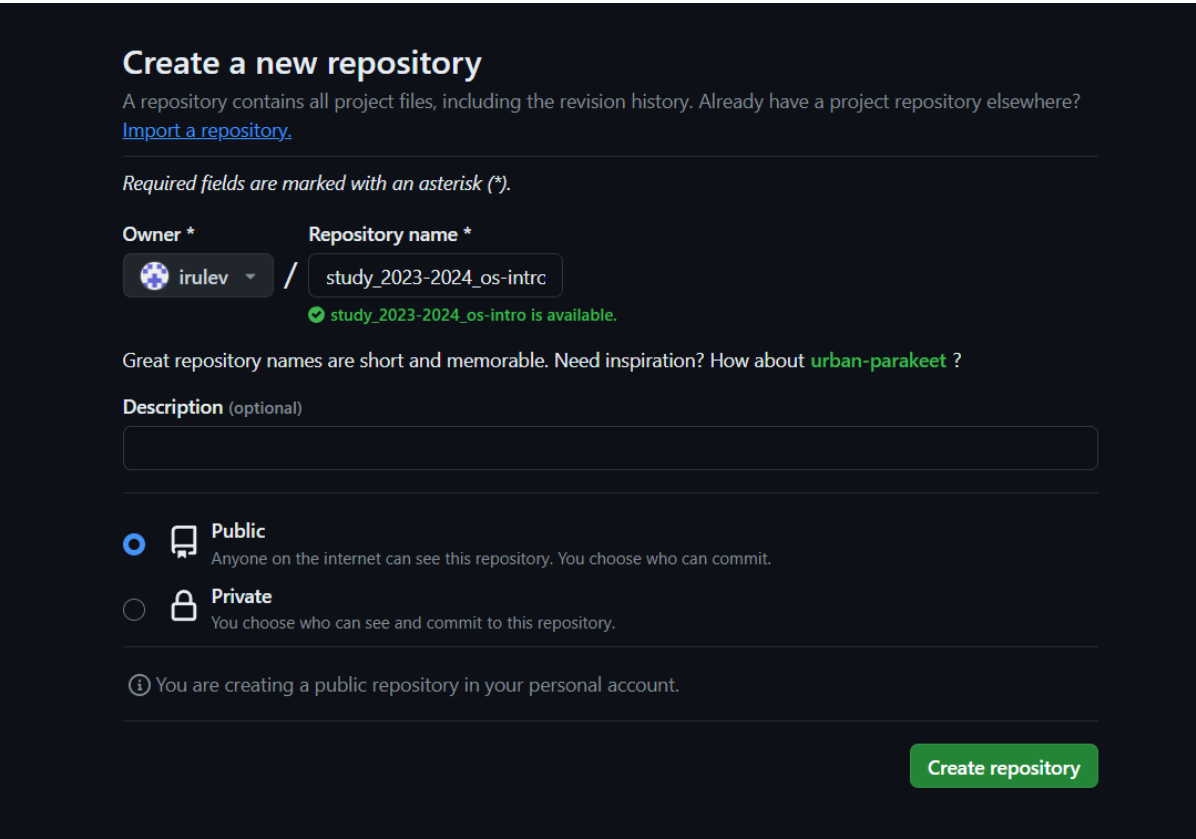
Add GPG key

Авторизируемся в Github с помощью команды `gh`



Создание рабочего пространства

Переходим в репозиторий с шаблоном и создаем из него шаблон.



После создания шаблона, склонируем репозиторий на локальную машину

```
Cloning into 'study_2023-2024_os-intro'...
remote: Enumerating objects: 32, done.
remote: Counting objects: 100% (32/32), done.
remote: Compressing objects: 100% (31/31), done.
remote: Total 32 (delta 1), reused 18 (delta 0), pack-reused 0
Receiving objects: 100% (32/32), 18.59 KiB | 346.00 KiB/s, done.
Resolving deltas: 100% (1/1), done.
[root@irulev ~]#
```

Перейдем в репозиторий и удалим `package.json`

Также создадим файл `COURSE` с текстом "os-intro"

```
[root@irulev ~]# cd ~/work/study/2022-2023/"Операционные системы"/os-intro
[root@irulev os-intro]# rm package.json
rm: remove regular file 'package.json'? y
[root@irulev os-intro]# rm package.json
rm: cannot remove 'package.json': No such file or directory
[root@irulev os-intro]# echo os-intro > COURSE
[root@irulev os-intro]# make
bash: make: command not found...
[root@irulev os-intro]# echo os-intro > COURSE
make
bash: make: command not found...
[root@irulev os-intro]# echo os-intro > COURSE make
```

Теперь запустим изменения. До этого исполнив `git add` и `git commit` с названием коммита

`feat(main): make course structure`

```
[root@irulev os-intro]# git push
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 6 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 281 bytes | 140.00 KiB/s, done.
Total 3 (delta 1), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To https://github.com/irulev/study_2023-2024_os-intro.git
   a7141f4..e24f944  master -> master
[root@irulev os-intro]#
```

Ответы на контрольные вопросы

Что такое системы контроля версий (VCS) и для решения каких задач они предназначены?

Системы контроля версий (VCS) предназначены для отслеживания изменений в программном коде и обеспечения коллективной разработки.

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

Хранилище: Место, где хранятся все изменения и версии программного кода.

Commit: Отдельное изменение или набор изменений в коде, зафиксированное в системе контроля версий.

История: Последовательность коммитов, отображающая эволюцию кода.

Рабочая копия: Локальная копия проекта, с которой работает разработчик.

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

Децентрализованные VCS копируют всю историю изменений на каждый клиентский компьютер, в то время как централизованные VCS хранят все изменения на центральном сервере и клиенты получают только последние версии файлов. Примеры децентрализованных VCS: Git, Mercurial. Примеры централизованных VCS: Subversion, CVS.

Опишите действия с VCS при единоличной работе с хранилищем.

При индивидуальной разработке пользователь клонирует проект на свой компьютер, вносит изменения и создает новые версии, коммитя их в системе контроля версий.

Опишите порядок работы с общим хранилищем VCS.

Пользователь получает версию проекта из центрального хранилища, вносит изменения, коммитит их и отправляет обратно в хранилище.

Каковы основные задачи, решаемые инструментальным средством git?

Git используется для разработки проектов в команде, контроля изменений в файлах и возможности сохранения нескольких состояний проекта.

Назовите и дайте краткую характеристику командам git.

`git add` - добавляет изменения для коммита.

`git commit` - сохраняет изменения в репозитории с названием.

`git push` - отправляет изменения на удаленный репозиторий.

`git config` - позволяет изменить настройки Git.

Приведите примеры использования при работе с локальным и удалённым репозиториями.

В локальном репозитории разработчик может вносить изменения в код и коммитить их без доступа к сети. В удаленном репозитории команда разработчиков может совместно работать над проектом, обмениваясь изменениями через централизованный сервер.

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

Ветви используются для параллельной разработки функций или исправлений, чтобы избежать конфликтов между изменениями и обеспечить безопасное тестирование нового кода.

Как и зачем можно игнорировать некоторые файлы при commit?

Файлы могут быть проигнорированы с помощью файла `.gitignore`, чтобы избежать загрязнения репозитория лишними или конфиденциальными файлами.

Выводы

Мы изучили идеологию применения средств контроля версий и освоили базовые команды git'a.