

Отчёт по лабораторной работе 2

Дисциплина Архитектура Компьютеров и Операционные Системы

Гомес Лопес Теофания

Содержание

1	Цель работы	5
2	Задание	6
3	Выполнение лабораторной работы	7
3.1	Создание базовой конфигурации для работы с git.	7
3.2	Создание ключ SSH:	8
3.3	Создание ключ gpg	8
3.4	Создание локального каталога для выполнения заданий.	12
4	Выводы	14
5	Ответы на контрольные вопросы	15
	Список литературы	18

Список иллюстраций

3.1	Установка git	7
3.2	Установка gh	7
3.3	Имя и email владельца	8
3.4	Имя начальной ветки и параметры	8
3.5	Создание ключ ssh	8
3.6	Создание ключ gpg	9
3.7	Настройки ключ gpg	9
3.8	личная информация	9
3.9	аккаунт на git	10
3.10	список ключей	10
3.11	Установка xclip	10
3.12	Копирование ключ gpg	10
3.13	Добавлен ключ gpg	11
3.14	указываю Git	11
3.15	авторизацию в gh	11
3.16	Авторизоваться через браузер	11
3.17	Завершена авторизация	12
3.18	Создание каталог	12
3.19	Создание каталог	12
3.20	Удаление файла	12
3.21	Созданы необходимых каталогов	13
3.22	Отправление файлы на сервер	13

Список таблиц

1 Цель работы

Изучение идеологии, применение средств контроля версий и освоение умения по работе с git.

2 Задание

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

3 Выполнение лабораторной работы

3.1 Создание базовой конфигурации для работы с git.

Устанавливаю git используя “dnf install git”:

```
teofaniagomeslopes@teofanialopes:~$ sudo -i
[sudo] senha para teofaniagomeslopes:
root@teofanialopes:~# dnf install git
Última verificação de metadados: 2:58:19 atrás em dom 02 mar 2025 15:13:38.
0 pacote git-2.48.1-1.fc40.x86_64 já está instalado.
Dependências resolvidas.
Nada para fazer.
Concluído!
root@teofanialopes:~#
```

Рис. 3.1: Установление git

С помощью dnf install gh, устанавливаю gh:

```
root@teofanialopes:~# dnf install gh
Última verificação de metadados: 2:59:56 atrás em dom 02 mar 2025 15:13:38.
Dependências resolvidas.
=====
Pacote      Arquitetura  Versão      Repositório  Tam.
=====
Instalando:
gh          x86_64       2.65.0-1.fc40  updates      11 M
Resumo da transação
=====
Instalar 1 pacote
```

Рис. 3.2: Установление gh

В качестве имя и email владельца репозитории задаю свои имя и email и настраиваю utf-8:

```
root@teofanialopes:~# git config --global user.name "ltgomes"
root@teofanialopes:~# git config --global user.email "lopesteofania3@gmail.com"
root@teofanialopes:~# git config --global core.quotepath false
root@teofanialopes:~#
```

Рис. 3.3: Имя и email владельца

Задаю имя начальной ветки и параметры autocrlf и safecrlf:

```
root@teofanialopes:~# git config --global init.defaultBranch master
root@teofanialopes:~# git config --global core.autocrlf input
root@teofanialopes:~# git config --global core.safecrlf warn
root@teofanialopes:~#
```

Рис. 3.4: Имя начальной ветки и параметры

3.2 Создание ключ SSH:

Создаю ключи ssh по алгоритму rsa с размером 4096 бит:

```
root@teofanialopes:~# ssh-keygen -t rsa -b 4096
Generating public/private rsa key pair.
Enter file in which to save the key (/root/.ssh/id_rsa):
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:kb4fY323qgmIIuQQ0jQSqN3YI/fH+M7NUNQMx6QK7Ao root@teofanialopes
The key's randomart image is:
+---[RSA 4096]-----+
|+.o      o.      |
|. + . . . . .o   |
|+..+  o o .+    |
|o.+ =. o o. o   |
|..Eo o.oS. .    |
|+ . . + +...    |
| o ... +.o+ . . .|
| . . .o=.o.. .. |
| .o.+o....      |
+---[SHA256]-----+
root@teofanialopes:~#
```

Рис. 3.5: Создание ключ ssh

3.3 Создание ключ gpg

Генерирую ключ gpg --full-generate-key:


```
root@teofania3:~# gpg --full-generate-key
gpg (GnuPG) 2.4.4; 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: pasta '/root/.gnupg' criada
Selecione o tipo de chave desejado:
(1) RSA e RSA
(2) DSA e Elgamal
(3) DSA (apenas de assinar)
(4) RSA (apenas de assinar)
(9) ECC (de assinar e cifrar) *pré-definição*
(10) ECC (apenas de assinar)
(14) Chave do cartão existente
Sua opção? 1
```

Рис. 3.6: Создание ключ gpg

Из предложенных опций выбираю тип RSA and RSA; размер 4096; срок действия 0:

```
Sua opção? 1
As chaves RSA podem estar entre 1024 e 4096 bits de comprimento.
Qual tamanho de chave você quer? (3072) 4096
O tamanho de chave pedido é 4096 bits
Especifique quando a chave expira.
  0 = chave não expira
  <n> = chave expira em n dias
  <n>w = chave expira em n semanas
  <n>m = chave expira em n meses
  <n>y = chave expira em n anos
Quando a chave expira? (0) 0
A chave não expira de forma alguma
Isto está correto? (s/N) s
```

Рис. 3.7: Настройки ключ gpg

GPG запросил личную информацию, которая сохранится в ключе Имя и адрес электронной почты:

```
O GnuPG precisa construir uma ID de utilizador para identificar sua chave.
Nome verdadeiro: ltgomes
Endereço de email: lopesteofania3@gmail.com
Comentário:
Você selecionou este USER-ID:
"ltgomes <lopesteofania3@gmail.com>"
```

Рис. 3.8: личная информация

У меня уже есть аккаунт на github, поэтому я вхожу в систему:

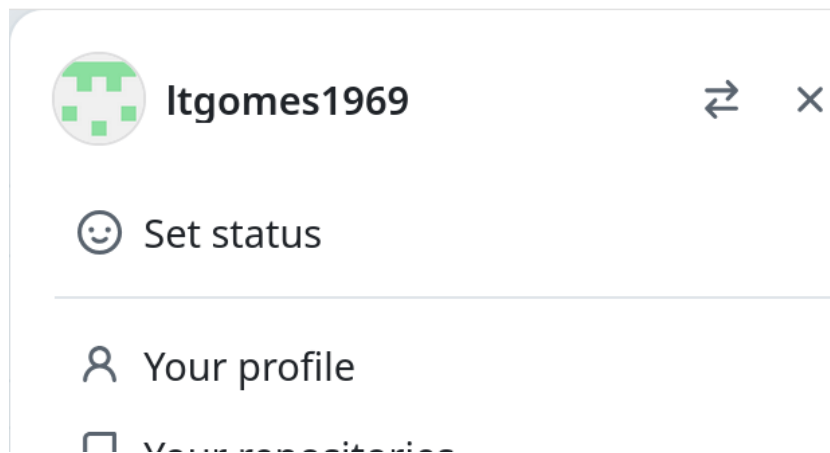


Рис. 3.9: аккаунт на git

Вывожу список ключей:

```
atividade (escrever no teclado, mover o rato, usar os discos) durante a
geração dos números primos; isto dá ao gerador de números aleatórios
uma hipótese maior de ganhar entropia suficiente.
Precisamos gerar muitos bytes aleatórios. É uma boa ideia realizar outra
atividade (escrever no teclado, mover o rato, usar os discos) durante a
geração dos números primos; isto dá ao gerador de números aleatórios
uma hipótese maior de ganhar entropia suficiente.
gpg: pasta '/home/teofaniagomeslopes/.gnupg/openpgp-revocs.d' criada
gpg: certificado de revogação armazenado como '/home/teofaniagomeslopes/.gnupg/o
penpgp-revocs.d/8BD1C625F51BBCBFC5AA211395FF0FD3E1D366AD.rev'
chaves pública e privada criadas e assinadas.

pub  rsa4096 2025-03-02 [SC]
      8BD1C625F51BBCBFC5AA211395FF0FD3E1D366AD
uid                               ltgomes <lopesteofania3@gmail.com>
sub  rsa4096 2025-03-02 [E]
```

Рис. 3.10: список ключей

Устанавливаю xclip:

```
teofaniagomeslopes@teofanialopes:~$ dnf install xclip
Erro: este comando deve ser executado com privilégios de superusuário (sob o usu
ário root na maioria dos sistemas).
```

Рис. 3.11: Установление xclip

Скопирую сгенерированный gpg ключ в буфер обмена:

```
teofaniagomeslopes@teofanialopes:~$ gpg --armor --export 95FF0FD3E1D366AD | xcli
p -sel clip
teofaniagomeslopes@teofanialopes:~$
```

Рис. 3.12: Копирование ключ gpg

Далее перехожу в настройки GitHub, нажимаю на кнопку New GPG key и вставляю полученный ключ:

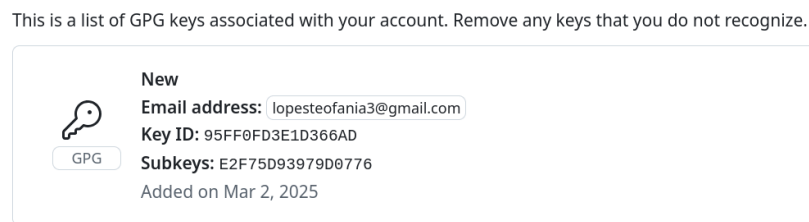


Рис. 3.13: Добавлен ключ gpg

Используя введённый email, указываю Git применять его при подписи коммитов:

```
teofaniagomeslopes@teofanialopes:~$ git config --global user.signinkey 95FF0FD3E1D366AD
teofaniagomeslopes@teofanialopes:~$ git config --global commit.gpgsign true
teofaniagomeslopes@teofanialopes:~$ git config --global gpg.program $(which gpg2)
teofaniagomeslopes@teofanialopes:~$
```

Рис. 3.14: указываю Git

Начинаю авторизацию в gh используя gh auth login:

```
teofaniagomeslopes@teofanialopes:~$ gh auth login
? Where do you use GitHub? GitHub.com
? What is your preferred protocol for Git operations on this host? SSH
? Upload your SSH public key to your GitHub account? /home/teofaniagomeslopes/.ssh/id_rsa.pub
? Title for your SSH key: GitHub CLI
? How would you like to authenticate GitHub CLI? Login with a web browser

! First copy your one-time code: 63F9-4DE1
Press Enter to open https://github.com/login/device in your browser...
```

Рис. 3.15: авторизацию в gh

Завершаю авторизацию на броузер:



Congratulations, you're all set!

Your device is now connected.

Рис. 3.16: Авторизоваться через броузер

```
! First copy your one-time code: 63F9-4DE1
Press Enter to open https://github.com/login/device in your browser...

✓ Authentication complete.
- gh config set -h github.com git_protocol ssh
✓ Configured git protocol
✓ SSH key already existed on your GitHub account: /home/teofaniagomeslopes/.ssh/
id_rsa.pub
✓ Logged in as ltgomes1969
```

Рис. 3.17: Завершена авторизация

3.4 Создание локального каталога для выполнения заданий.

Создаю каталог “mkdir -p ~/work/study/2022-2023/”Операционные системы”:

```
teofaniagomeslopes@teofanialopes:~$ mkdir -p ~/work/study/2024-2025/"Операционные системы"
teofaniagomeslopes@teofanialopes:~$ cd ~/work/study/2024-2025/"Операционные системы"
teofaniagomeslopes@teofanialopes:~/work/study/2024-2025/Операционные системы$
```

Рис. 3.18: Создание каталог

Перехожу в созданный каталог:

```
teofaniagomeslopes@teofanialopes:~$ gh repo create study_2024-2025_os-intro --te
mplate=yamadharma/course-directory-student-template --public
✓ Created repository ltgomes1969/study_2024-2025_os-intro on GitHub
https://github.com/ltgomes1969/study_2024-2025_os-intro
teofaniagomeslopes@teofanialopes:~$ git clone --recursive git@github.com:ltgomes
1969/study_2024-2025_os-intro.git os-intro
Cloning into 'os-intro'...
ERROR: Repository not found.
fatal: Could not read from remote repository.

Please make sure you have the correct access rights
and the repository exists.
teofaniagomeslopes@teofanialopes:~$ git clone --recursive git@github.com:ltgomes
1969/study_2024-2025_os-intro.git os-intro
Cloning into 'os-intro'...
remote: Enumerating objects: 36, done.
remote: Counting objects: 100% (36/36), done.
remote: Compressing objects: 100% (35/35), done.
remote: Total 36 (delta 1), reused 21 (delta 0), pack-reused 0 (from 0)
```

Рис. 3.19: Создание каталог

Удаляю лишние файлы:

```
teofaniagomeslopes@teofanialopes:~$ cd os-intro
teofaniagomeslopes@teofanialopes:~/os-intro$ rm package.json
teofaniagomeslopes@teofanialopes:~/os-intro$ echo os-intro > COURSE
```

Рис. 3.20: Удаление файла

Создаю еще необходимые каталоги:

```
teofaniagomeslopes@teofanialopes:~/os-intro$ make
Usage:
  make <target>

Targets:
  list           List of courses
  prepare       Generate directories structure
  submodule     Update submules
```

Рис. 3.21: Создани необходимых каталогов

Отправляю Файлы на сервер:

```
teofaniagomeslopes@teofanialopes:~/os-intro$ git add .
teofaniagomeslopes@teofanialopes:~/os-intro$ git commit -am 'feat(main): make co
urse structure'
[master 9fa3129] feat(main): make course structure
 2 files changed, 1 insertion(+), 14 deletions(-)
 delete mode 100644 package.json
teofaniagomeslopes@teofanialopes:~/os-intro$ git push
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 4 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 948 bytes | 948.00 KiB/s, done.
Total 3 (delta 1), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To github.com:ltgomes1969/study_2024-2025_os-intro.git
   afeca9b..9fa3129  master -> master
teofaniagomeslopes@teofanialopes:~/os-intro$
```

Рис. 3.22: Отправление файлы на сервер

4 Выводы

При выполнении лабораторной работы я изучила идеологию, применение средств контроля версий и освоила умение по работе с git.

5 Ответы на контрольные вопросы

1. Системы Контроля Версий - Программные инструменты, помогающие командам разработчиков управлять изменениями в исходном коде с течением времени.
2. Хранилище - в нем хранятся все документы, включая историю их изменение и прочей служебной информацией.

commit - отслеживание изменений сохраняет разницу в изменениях.

история - Хранит все изменения в проекте и позволяет при необходимости обратиться к нужным данным.

рабочая копия- копия проекта основанная на версии из хранилища.
3. В централизованном VCS например AccuRev, каждый пользователь копирует себе необходимые ему файлы из репозитория, изменяет их а затем добавляет изменения обратно в хранилище. В децентрализованном VCS например Git, есть возможность добавлять и забирать изменения из любого репозитория.
4. Сначала создается и подключается удаленный репозиторий, затем по мере изменения проекта эти изменения отправляются на сервер.
5. Участник проекта перед началом работы получает нужную ему версию проекта в хранилище, с помощью определенных команд, после внесения изменений пользователь размещает новую версию в хранилище. При этом предыдущие версии не удаляются. К ним можно вернуться в любой момент.

6. Хранение информации о всех изменениях в вашем коде, обеспечение удобства командной работы над кодом.

Создание основного дерева репозитория: `git init`

7. 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 изменений: `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. `git push -all` отправляем из локального репозитория все сохраненные изменения в центральный репозиторий, предварительно создав локальный репозиторий и сделав предварительную конфигурацию.
9. Ветвление - один из параллельных участков в одном хранилище, исходящих из одной версии, обычно есть главная ветка. Между ветками, т. е. их концами возможно их слияние. Используются для разработки новых функций.
10. Во время работы над проектом могут создаваться файлы, которые не следуют добавлять в репозиторий. Например, временные файлы. Можно прописать шаблоны игнорируемых при добавлении в репозиторий типов файлов в файл `.gitignore` с помощью сервисов.

Список литературы