Лабораторная работа № 5

Отчёт

Ермишина Мария Кирилловна

Содержание

# 1 Цель работы

Целью данной лабораторной работы является настройка рабочей среды. [1]

# 2 Теоретическое введение

Менеджер паролей pass — программа, сделанная в рамках идеологии Unix. Также носит название стандартного менеджера паролей для Unix (The standard Unix password manager).

Основные свойства: - Данные хранятся в файловой системе в виде каталогов и файлов. - Файлы шифруются с помощью GPG-ключа.

# 3 Выполнение лабораторной работы

1. Менеджер паролей pass Для начала мы устанавливаем pass и gopass в нашей виртуальной машине. Делаем это с помощью следующих команд: (рис. 1)

* dnf install pass pass-otp
* dnf install gopass

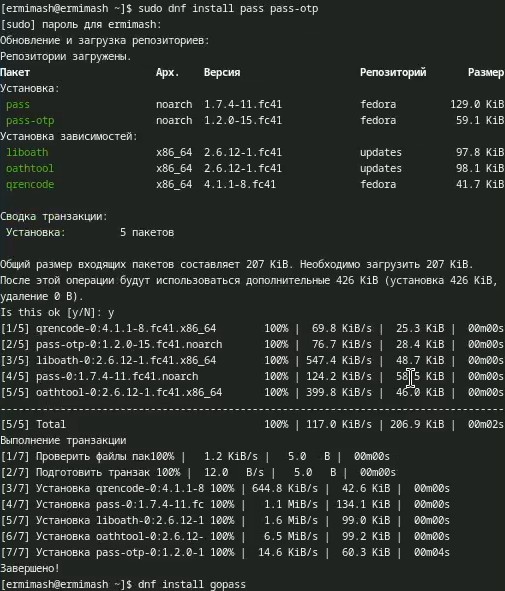


Рис. 1: Установка менеджера

После установки необходима настройка. (рис. 2) - Первым шагом проверяем (при необходимости создаём) ключ GPG. Наличие ключей проверяем с помощью команды gpg –list-secret-keys. - Инициализируем хранилище: pass init 1132230166@pfur.ru - Синхронизируем структуру с git: - pass git init - pass git remote add origin git@github.com:ErmiMash/pass.git (создаём репозиторий заранее) - pass git pull - pass git push - Вручную закоммитим и выложим изменения: (рис. 3) - cd ~/.password-store/ - git add . - git commit -am ‘edit manually’ - git push - pass git status (проверяем статус синхронизации)



Рис. 2: Настройка

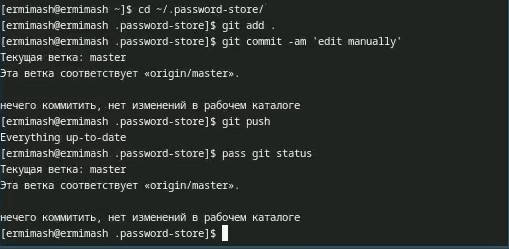


Рис. 3: Синхронизация с git

1. Настройка интерфейса с броузером Устанавливаем программу, обеспечивающую интерфейс native messaging, с помощью команд: (рис. 4)

* dnf copr enable maximbaz/browserpass
* dnf install browserpass

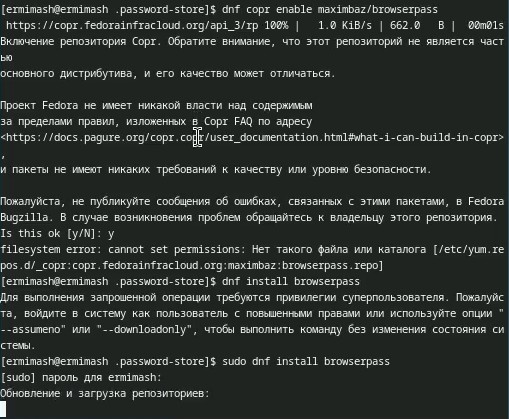


Рис. 4: Установка программы

После этого устанавливаем плагин на наш браузер FireFox, перейдя по ссылке https://addons.mozilla.org/en-US/firefox/addon/browserpass-ce/. (рис. 5)

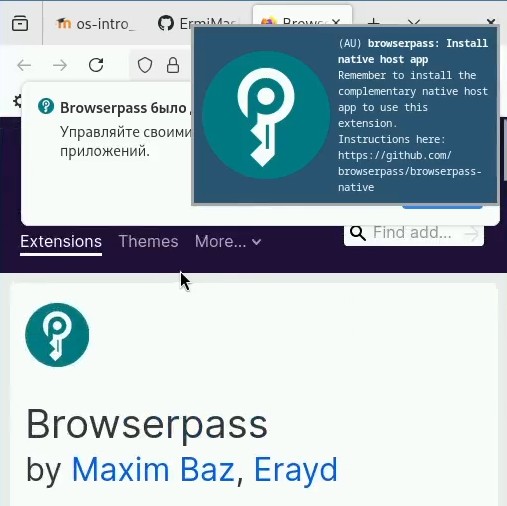


Рис. 5: Установка плагина

1. Сохранение пароля Для начала добавляем новый пароль с помощью команды pass insert pass.txt. До этого создаём сам файл pass.txt. (рис. 6)

Отображаем наш пароль, после чего заменяем его. Делаем это с помощью следующих команд: (рис. 6) - pass pass.txt - pass generate –in-place pass.txt

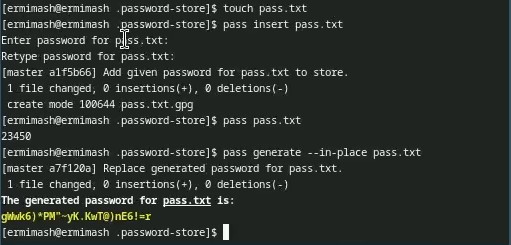


Рис. 6: Сохранение пароля

1. Управление файлами конфигурации Установливаем дополнительное программное обеспечение: (рис. 7)

* sudo dnf -y install  
  dunst  
  fontawesome-fonts  
  powerline-fonts  
  light  
  fuzzel  
  swaylock  
  kitty  
  waybar swaybg  
  wl-clipboard  
  mpv  
  grim  
  slurp

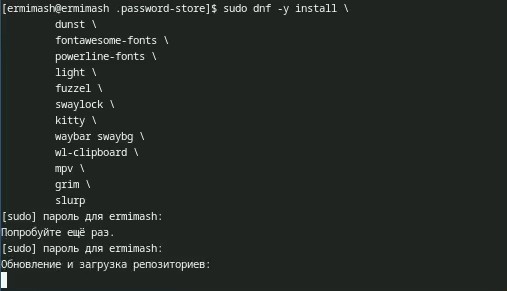


Рис. 7: Установка доп. прог. обеспеч.

Устанавливаем шрифты: (рис. 8) - sudo dnf copr enable peterwu/iosevka - sudo dnf search iosevka - sudo dnf install iosevka-fonts iosevka-aile-fonts iosevka-curly-fonts iosevka-slab-fonts iosevka-etoile-fonts iosevka-term-fonts

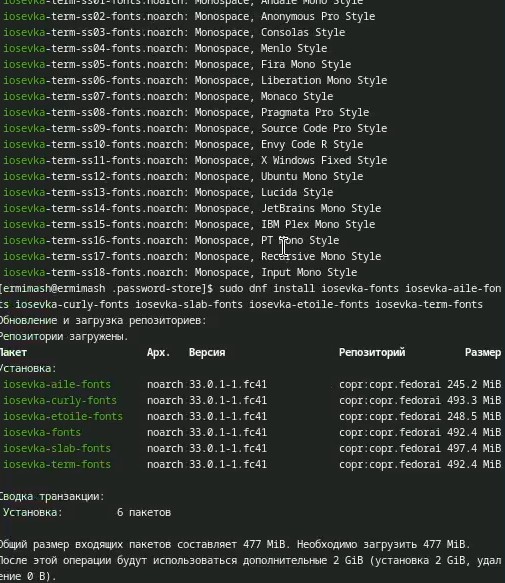


Рис. 8: Установка шрифтов

1. Установка бинарного файла Скрипт определяет архитектуру процессора и операционную систему и скачивает необходимый файл. Устанавливаем: (рис. 9)

* sh -c “$(wget -qO- chezmoi.io/get)”

1. Создание собственного репозитория с помощью утилит Будем использовать утилиты командной строки для работы с github. Создадим свой репозиторий для конфигурационных файлов на основе шаблона: (рис. 9)

* gh repo create dotfiles –template=“yamadharma/dotfiles-template” –private

1. Подключение репозитория к своей системе Инициализируем chezmoi с репозиторием dotfiles: (рис. 9)

* chezmoi init git@github.com:ErmiMash/dotfiles.git

После проверяем, какие изменения внесёт chezmoi в домашний каталог (chezmoi diff), и, если они нас устраивают, запускаем chezmoi apply -v. (рис. 10)

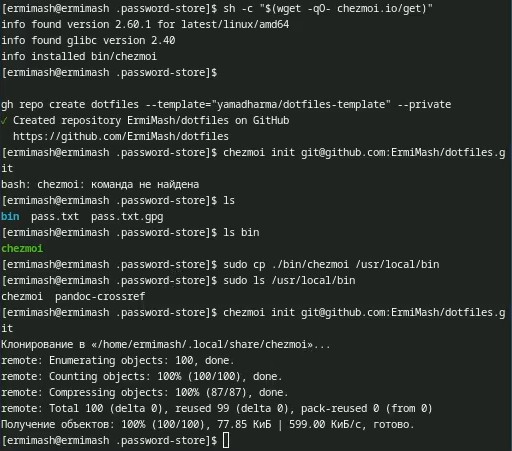


Рис. 9: Работа с репозиторием

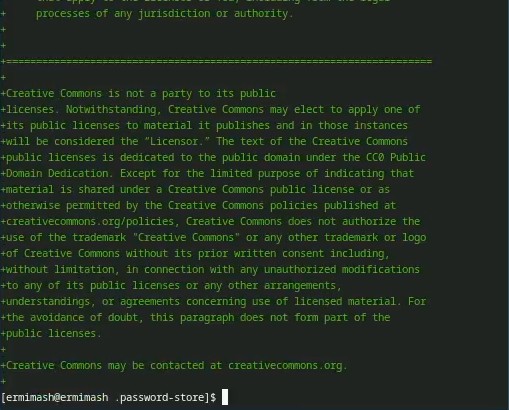


Рис. 10: Проверка изменений

1. Использование chezmoi на нескольких машинах Воспользуеммся второй виртуальной машиной - в моём случае это Ubuntu. На второй машине инициализируйте chezmoi с вашим репозиторием dotfiles через SSH: рис. 11)

* chezmoi init git@github.com:ErmiMash/dotfiles.git

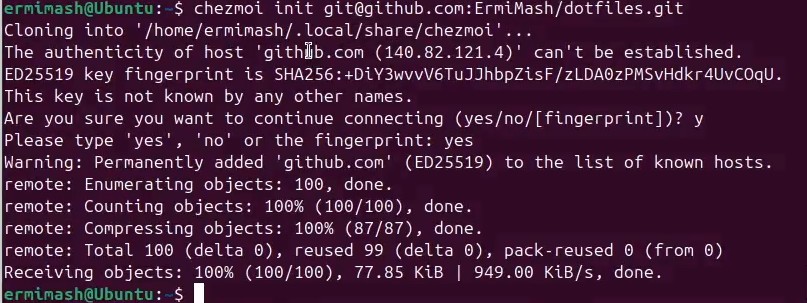


Рис. 11: Запуск chezmoi на второй в.м.

После этого так же проверяем, какие изменения внесёт chezmoi в домашний каталог (chezmoi diff), и, если они нас устраивают, запускаем chezmoi apply -v.

Нас устраивают изменения, поэтому редактировать и оъединять данные не нужно.

При существующем каталоге chezmoi можно получить и применить последние изменения из вашего репозитория: (рис. 12) - chezmoi update -v

1. Настройка новой машины с помощью одной команды Можно установить свои dotfiles на новый компьютер с помощью одной команды: (рис. 12)

* chezmoi init –apply git@github.com:ErmiMash/dotfiles.git

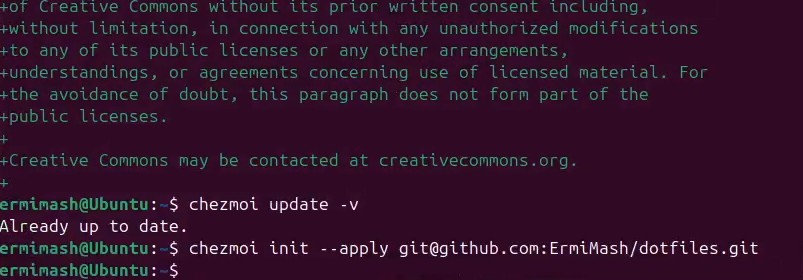


Рис. 12: Настройка каталогов

1. Ежедневные операции c chezmoi Извлекаем последние изменения из репозитория и примените их с помощью команды chezmoi update (это запускает git pull –autostash –rebase в вашем исходном каталоге, а затем chezmoi apply). (рис. 13)

Извлекаем последние изменения из своего репозитория и посмотрите, что изменится, фактически не применяя изменения: chezmoi git pull – –autostash –rebase && chezmoi diff (это запускается git pull –autostash –rebase в вашем исходном каталоге, а chezmoi diff затем показывает разницу между целевым состоянием, вычисленным из вашего исходного каталога, и фактическим состоянием). (рис. 13)

Если вы довольны изменениями - chezmoi apply. (рис. 13)

Автоматически фиксируем и отправляем изменения в репозиторий - функцию нужно подключить, так как она отключена по умолчанию. Чтобы включить её, добавьте в файл конфигурации ~/.config/chezmoi/chezmoi.toml следующее: (рис. 14)

* [git] autoCommit = true autoPush = true

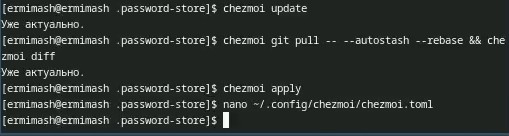


Рис. 13: Работа с репозиторием

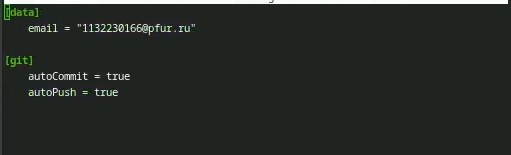


Рис. 14: Редакция файла

# 4 Выводы

В ходе данной лабораторной работы получила навыки настройки рабочей среды.

# Список литературы

1. Д. К. [Лабораторная работа № 5](https://esystem.rudn.ru/mod/page/view.php?id=1224101#orgb06553f). RUDN.