Отчёт по лабораторной работе №5

Настройка рабочей среды

Чекмарев Александр Дмитриевич | Группа НПИбд-02-23

Содержание

# 1 Цель работы

Получение навыков работы с pass и chezmoi

# 2 Выполнение лабораторной работы

## 2.1 Менеджер паролей pass

### 2.1.1 Установка

Установим pass

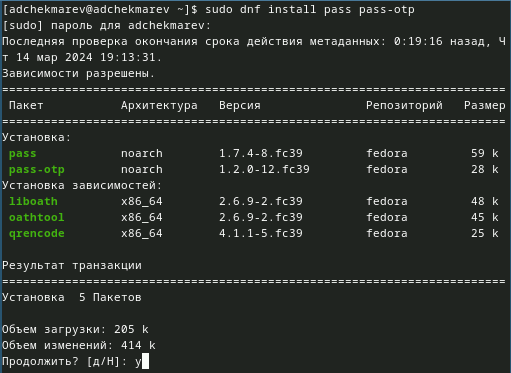


Рис 1.1.1: установка pass

Установим gopass

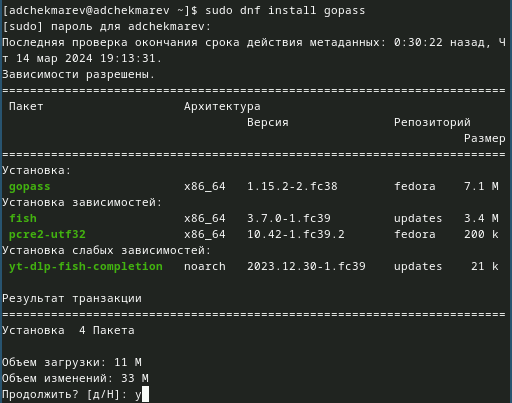


Рис 1.1.2: установка gopass

### 2.1.2 Настройка

Посмотрим список ключей: gpg –list-secret-keys

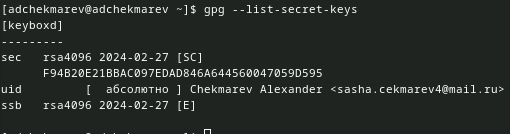


Рис 1.2.1: просмотр ключей

Если ключа нет, то можно создать новый (мне не нужно):  
gpg –full-generate-key

Инициализируем хранилище и создадим структуру:

pass init   
pass git init

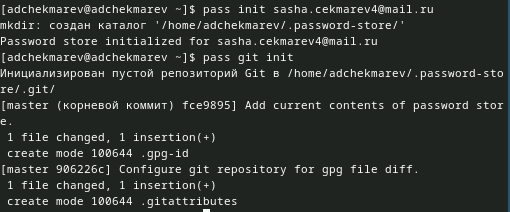


Рис 1.2.2: инициализация хранилища и создание структуры

Также можно задать адрес репозитория на хостинге (репозиторий необходимо предварительно создать). Создадим репозиторий на сайте гитхаб:

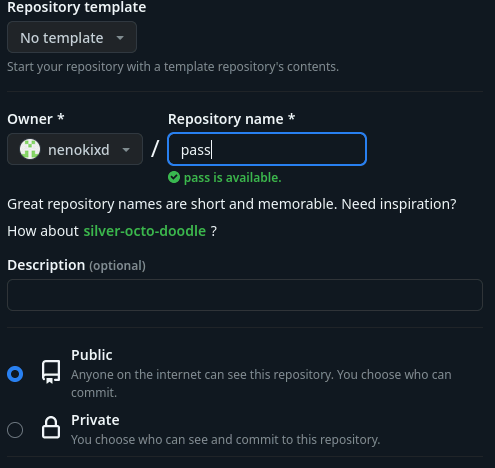


Рис 1.2.3: создание репозитория

Зададим адрес репозитория на хостинге:

pass git remote add origin git@github.com:/.git

Рис 1.2.4: адрес репозитория

Рис 1.2.4: адрес репозитория

Для синхронизации выполним следующие команда:

pass git pull

Рис 1.2.5: проверка файлов на актуальность

Рис 1.2.5: проверка файлов на актуальность

pass git push

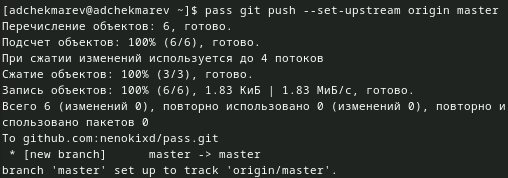


Рис 1.2.6: отправка

**Прямые изменения**

Следует заметить, что отслеживаются только изменения, сделанные через сам gopass (или pass). Если изменения сделаны непосредственно на файловой системе, необходимо вручную закоммитить и выложить изменения:

cd ~/.password-store/  
git add .  
git commit -am ‘edit manually’  
git push

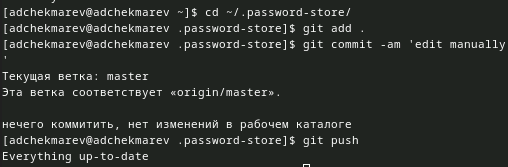


Рис 1.2.7: коммит

Проверить статус синхронизации можно командой

pass git status

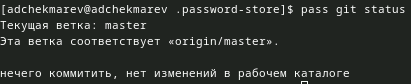


Рис 1.2.8: проверка статуса

### 2.1.3 Настройка интерфейса с браузером

Для взаимодействия с браузером используется интерфейс native messaging. Поэтому кроме плагина к браузеру устанавливается программа, обеспечивающая интерфейс native messaging. Плагин browserpass для Firefox: https://addons.mozilla.org/en-US/firefox/addon/browserpass-ce/.

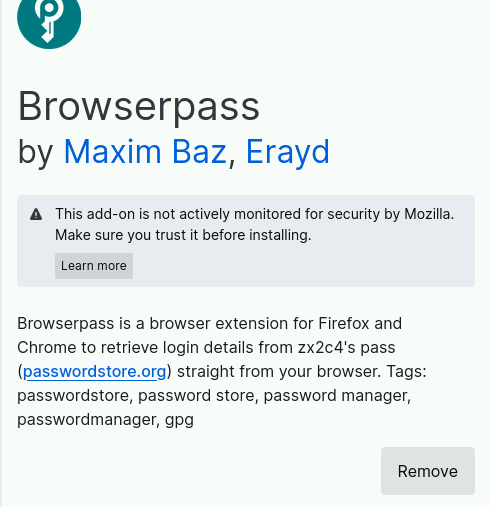


Рис 1.3.1: установка расширения

Интерфейс для взаимодействия с браузером (native messaging)

Репозиторий: https://github.com/browserpass/browserpass-native

dnf copr enable maximbaz/browserpass

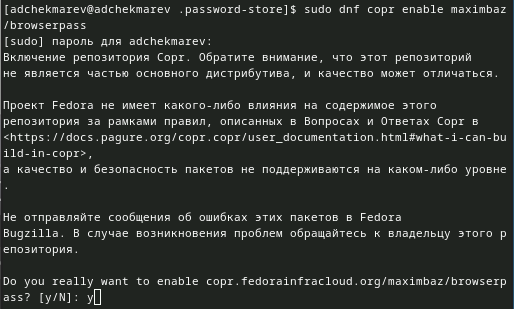


Рис 1.3.2: включение репозитория

dnf install browserpass

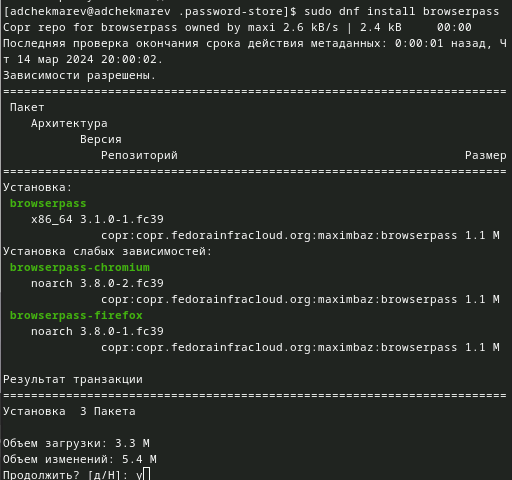


Рис 1.3.3: установка browserpass

### 2.1.4 Сохранение пароля

Добавить новый пароль

Выполним:

pass insert [OPTIONAL DIR]/[FILENAME]

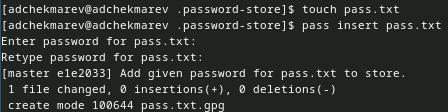


Рис 1.4.1: добавление пароля для файла

OPTIONAL DIR: необязательное имя каталога, определяющее файловую структуру для вашего хранилища паролей; FILENAME: имя файла, который будет использоваться для хранения пароля. Отобразим пароль для указанного имени файла:

pass [OPTIONAL DIR]/[FILENAME]

Рис 1.4.2: отображение пароля

Рис 1.4.2: отображение пароля

Заменим существующий пароль:

pass generate –in-place FILENAME

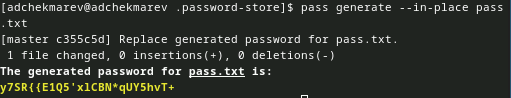


Рис 1.4.3: замена пароля

## 2.2 Управление файлами конфигурации

### 2.2.1 Дополнительное программное обеспечение

Установим дополнительное программное обеспечение:

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

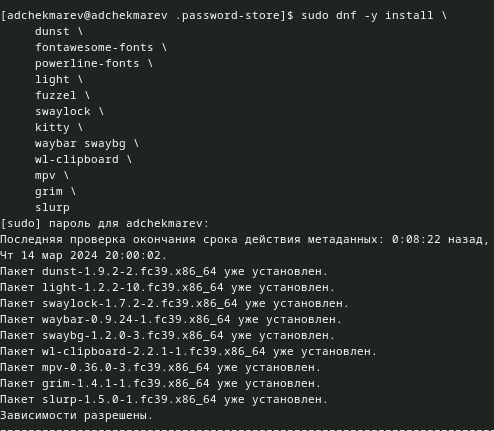


Рис 2.1.1: установка программ

Установим шрифты:

sudo dnf copr enable peterwu/iosevka

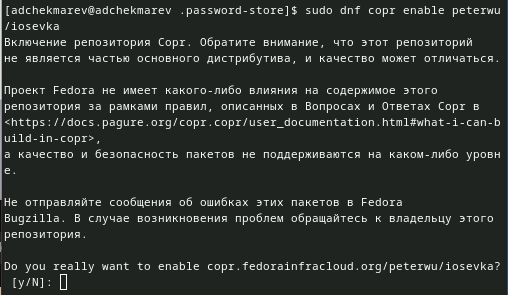


Рис 2.1.2: включение репозитория

sudo dnf search iosevka

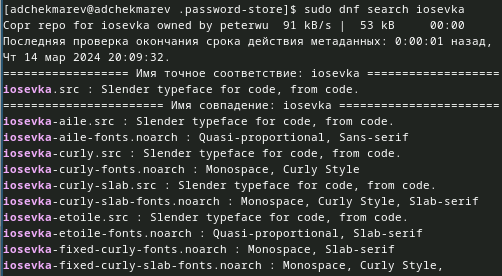


Рис 2.1.3: установка шрифтов

sudo dnf install iosevka-fonts iosevka-aile-fonts iosevka-curly-fonts iosevka-slab-fonts iosevka-etoile-fonts iosevka-term-fonts

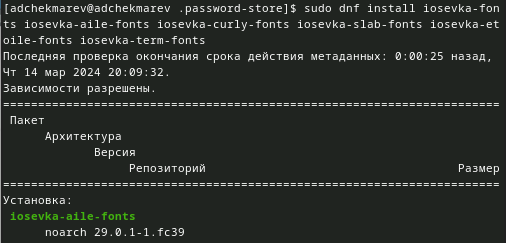


Рис 2.1.4: установка

### 2.2.2 Установка

Установка бинарного файла. Скрипт определяет архитектуру процессора и операционную систему и скачивает необходимый файл: с помощью wget:

sh -c “$(wget -qO- chezmoi.io/get)”

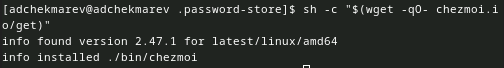


Рис 2.2.1: установка бинарного файла

### 2.2.3 Создание собственного репозитория с помощью утилит

Будем использовать утилиты командной строки для работы с github. Создадим свой репозиторий для конфигурационных файлов на основе шаблона:

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

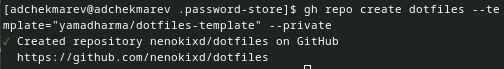


Рис 2.3.1: создание репозитория

### 2.2.4 Подключение репозитория к своей системе

Инициализируем chezmoi с нашим репозиторием dotfiles:

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

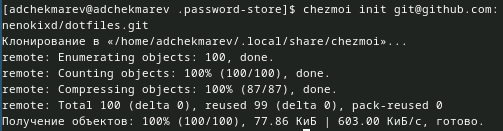


Рис 2.4.1: инициализация

Проверим, какие изменения внесёт chezmoi в домашний каталог, запустив:

chezmoi diff

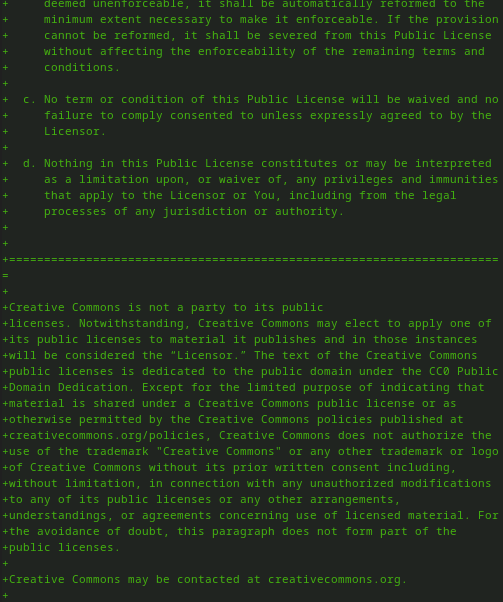


Рис 2.4.2: фрагмент проверки

Если нас устраивают изменения, внесённые chezmoi, запустим:

chezmoi apply -v

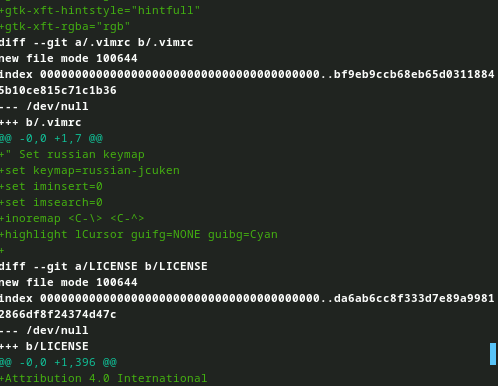


Рис 2.4.3: фрагмент внесения изменений

### 2.2.5 Использование chezmoi на нескольких машинах

На второй машине инициализируйте chezmoi с репозиторием dotfiles:

chezmoi init https://github.com//dotfiles.git

Или через ssh:

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

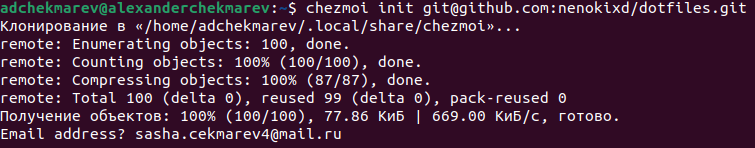


Рис 2.5.1: инициализация

Проверим, какие изменения внесёт chezmoi в домашний каталог, запустив:

chezmoi diff

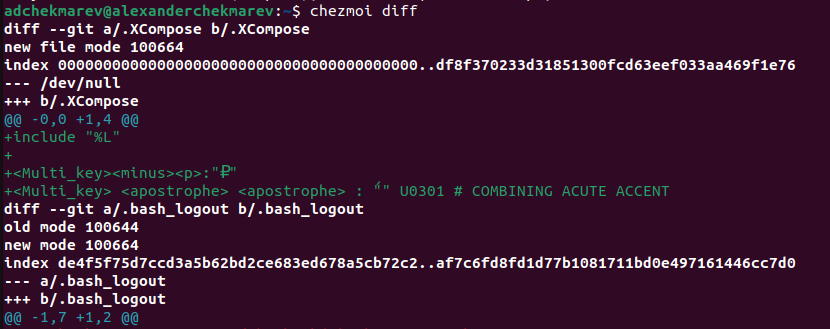


Рис 2.5.2: проверка

Если вас устраивают изменения, внесённые chezmoi, запустите:

chezmoi apply -v

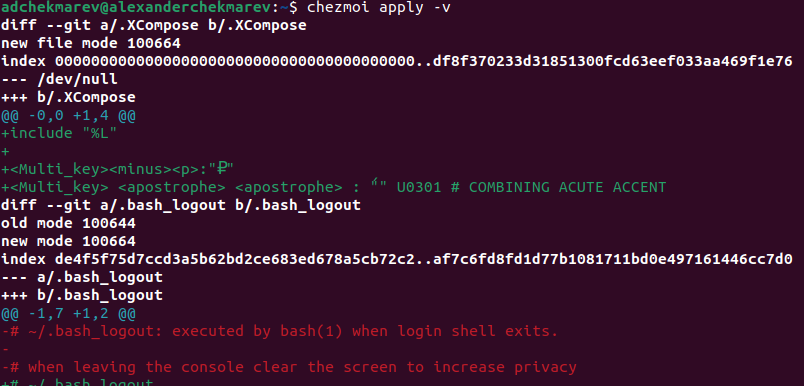


Рис 2.5.3: применим изменения

Если вас не устраивают изменения в файле, отредактируйте его с помощью:

chezmoi edit file\_name

Также можно вызвать инструмент слияния, чтобы объединить изменения между текущим содержимым файла, файлом в вашей рабочей копии и измененным содержимым файла:

chezmoi merge file\_name

При существующем каталоге chezmoi можно получить и применить последние изменения из вашего репозитория:

chezmoi update -v

Рис 2.5.4: применим последние изменения

Рис 2.5.4: применим последние изменения

### 2.2.6 Настройка новой машины с помощью одной команды

Можно установить свои dotfiles на новый компьютер с помощью одной команды:

chezmoi init –apply https://github.com//dotfiles.git

Через ssh:

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

Рис 2.6.1: устнаовка dotfiles

Рис 2.6.1: устнаовка dotfiles

## 2.3 Ежедневные операции c chezmoi

1. Извлечем последние изменения из репозитория и примените их

Можно извлечь изменения из репозитория и применить их одной командой:

chezmoi update

Рис 3.1.1: проверка актуальности файлов

Рис 3.1.1: проверка актуальности файлов

Это запускается git pull –autostash –rebase в вашем исходном каталоге, а затем chezmoi apply.

1. Извлечем последние изменения из своего репозитория и посмотрим, что изменится, фактически не применяя изменения

Выполним:

chezmoi git pull – –autostash –rebase && chezmoi diff

Рис 3.1.2: извлечение последних изменений

Рис 3.1.2: извлечение последних изменений

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.

Если мы довольны изменениями, то можем применить их:

chezmoi apply

Рис 3.1.3: применение изменений

Рис 3.1.3: применение изменений

1. Автоматически фиксируйте и отправляйте изменения в репозиторий

Можно автоматически фиксировать и отправлять изменения в исходный каталог в репозиторий. Эта функция отключена по умолчанию. Чтобы включить её, добавим в файл конфигурации ~/.config/chezmoi/chezmoi.toml следующее:

[git]  
autoCommit = true  
autoPush = true

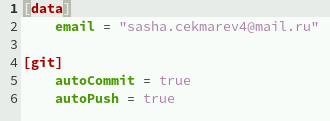


Рис 3.1.4: демонстрация файла

Всякий раз, когда в исходный каталог вносятся изменения, chezmoi фиксирует изменения с помощью автоматически сгенерированного сообщения фиксации и отправляет их в ваш репозиторий. Будьте осторожны при использовании autoPush. Если ваш репозиторий dotfiles является общедоступным, и вы случайно добавили секрет в виде обычного текста, этот секрет будет отправлен в ваш общедоступный репозиторий.

# 3 Выводы

Я научился пользоваться pass и chezmoi

# Список литературы