

Лабораторная работа №5

Операционные системы

Чувакина М. В.

14 марта 2024

Российский университет дружбы народов, Москва, Россия

Информация

- Чувакина Мария Владимировна, НКАбд-06-23
- Студентка факультета физико-математических и естественных наук
- Российский университет дружбы народов
- 1132236055@rudn.ru
- <https://mvchuvakina.github.io/ru/>

Цель данной лабораторной работы - настройка рабочей среды.

Менеджер паролей pass

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.

Структура базы паролей

Структура базы может быть произвольной, если Вы собираетесь использовать её напрямую, без промежуточного программного обеспечения. Тогда семантику структуры базы данных Вы держите в своей голове. Если же необходимо использовать дополнительное программное обеспечение, необходимо семантику заложить в структуру базы паролей.

Управление файлами конфигурации

Использование `chezmoi` для управления файлами конфигурации домашнего каталога пользователя.

Синтаксис шаблона

Действия шаблона записываются внутри двойных фигурных скобок, `{{ }}`. Действия могут быть переменными, конвейерами или операторами управления. Текст вне действий копируется буквально.

Выполнение лабораторной работы

Установка с помощью команд `pass dnf install pass` `pass-otp gopass dnf install gopass`

Ключи GPG

Просмотр списка ключей:

```
gpg --list-secret-keys
```

Если ключа нет, нужно создать новый:

```
gpg --full-generate-key
```

Инициализируем хранилище:

```
pass init
```

Создадим структуру git:

```
pass git init
```

Также можно задать адрес репозитория на хостинге (репозиторий необходимо предварительно создать):

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

Для синхронизации выполняется следующая команда:

```
pass git pull pass git push
```

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
```

Проверить статус синхронизации можно командой

```
pass git status
```

Для взаимодействия с браузером используется интерфейс `native messaging`. Поэтому кроме плагина к браузеру устанавливается программа, обеспечивающая интерфейс `native messaging`.

Репозиторий: <https://github.com/browserpass/browserpass-extension> Плагин для броузера
Плагин для Firefox: <https://addons.mozilla.org/en-US/firefox/addon/browserpass-ce/>. Плагин
для Chrome/Chromium: <https://chrome.google.com/webstore/detail/browserpass-ce/naepdomgkenhinolocfifgehiddafch>.

Сохранение пароля

Добавить новый пароль

Выполните:

```
pass insert [OPTIONAL DIR]/[FILENAME]
```

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

Отобразите пароль для указанного имени файла:

```
pass [OPTIONAL DIR]/[FILENAME]
```

Замените существующий пароль:

```
pass generate --in-place FILENAME
```

Дополнительное программное обеспечение

Установите дополнительное программное обеспечение:

```
sudo dnf -y install
```

```
dunst
```

```
fontawesome-fonts
```

```
powerline-fonts
```

```
light
```

```
fuzzel
```

```
swaylock
```

```
kitty
```

```
waybar swaybg
```

```
wl-clipboard
```

```
mpv
```

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

с помощью wget:

```
sh -c "$(wget -qO- chezmoi.io/get)"
```

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
```

Подключение репозитория к своей системе

Инициализируйте chezmoi с вашим репозиторием dotfiles:

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

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

```
chezmoi diff
```

Если вас устраивают изменения, внесённые chezmoi, запустите:

```
chezmoi apply -v
```

Использование chezmoi на нескольких машинах

На второй машине инициализируйте chezmoi с вашим репозиторием dotfiles:

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

Или через ssh:

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
```

Извлеките последние изменения из репозитория и примените их

Можно извлечь изменения из репозитория и применить их одной командой:

```
chezmoi update
```

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

Извлеките последние изменения из своего репозитория и посмотрите, что изменится, фактически не применяя изменения

Выполните:

```
chezmoi git pull --autostash --rebase && chezmoi diff
```

Я научилась настраивать рабочую среду.

...