Лабораторная работа №5

Операционные системы

Чувакина Мария Владимировна

Содержание

Список иллюстраций

Список таблиц

1 Цель работы

Цель данной лабораторной работы - настройка рабочей среды.

2 Задание

1. Менеджер паролей pass
2. Установка
3. Настройка
4. Настройка интерфейса с броузером
5. Сохранение пароля
6. Управление файлами конфигурации
7. Дополнительное программное обеспечение
8. Установка
9. Создание собственного репозитория с помощью утилит
10. Подключение репозитория к своей системе
11. Использование chezmoi на нескольких машинах
12. Настройка новой машины с помощью одной команды
13. Ежедневные операции с chezmoi

3 Теоретическое введение

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.

Семантическая структура базы паролей

Рассмотрим пользователя user в домене example.com, порт 22.

Отсутствие имени пользователя или порта в имени файла означает, что любое имя пользователя и порт будут совпадать:

```
example.com.pgp
```

Соответствующее имя пользователя может быть именем файла внутри каталога, имя которого совпадает с хостом. Это полезно, если в базе есть пароли для нескольких пользователей на одном хосте:

```
example.com/user.pgp
```

Имя пользователя также может быть записано в виде префикса, отделенного от хоста знаком @:

```
user@example.com.pgp
```

Соответствующий порт может быть указан после хоста, отделённый двоеточием (:):

```
example.com:22.pgp
example.com:22/user.pgp
user@example.com:22.pgp
```

Эти все записи могут быть расположены в произвольных каталогах, задающих Вашу собственную иерархию.

Управление файлами конфигурации

Использование chezmoi для управления файлами конфигурации домашнего каталога пользователя.

Конфигурация chezmoi

Рабочие файлы

Состояние файлов конфигурации сохраняется в каталоге

```
~/.local/share/chezmoi
```

Он является клоном вашего репозитория dotfiles. Файл конфигурации ~/.config/chezmoi/chezmoi.toml (можно использовать также JSON или YAML) специфичен для локальной машины. Файлы, содержимое которых одинаково на всех ваших машинах, дословно копируются из исходного каталога. Файлы, которые варьируются от машины к машине, выполняются как шаблоны, обычно с использованием данных из файла конфигурации локальной машины для настройки конечного содержимого, специфичного для локальной машины.

При запуске

```
chezmoi apply
```

вычисляется желаемое содержимое и разрешения для каждого файла, а затем вносит необходимые изменения, чтобы ваши файлы соответствовали этому состоянию.

```
По умолчанию chezmoi изменяет файлы только в рабочей копии.
```

Автоматически создавать файл конфигурации на новой машине

При выполнении chezmoi init также может автоматически создать файл конфигурации, если он еще не существует. Если ваш репозиторий содержит файл с именем .chezmoi.$FORMAT.tmpl, где $FORMAT есть один из поддерживаемых форматов файла конфигурации (json, toml, или yaml), то chezmoi init выполнит этот шаблон для создания исходного файла конфигурации.

Пересоздание файл конфигурации

Если вы измените шаблон файла конфигурации, chezmoi предупредит вас, если ваш текущий файл конфигурации не был сгенерирован из этого шаблона.

Вы можете повторно сгенерировать файл конфигурации, запустив:

```
chezmoi init
```

Способы создания файла шаблона

При первом добавлении файла передайте аргумент –template:

```
chezmoi add --template ~/.zshrc
```

Если файл уже контролируется chezmoi, но не является шаблоном, можно сделать его шаблоном:

```
chezmoi chattr +template ~/.zshrc
```

Можно создать шаблон вручную в исходном каталоге, присвоив ему расширение .tmpl:

```
chezmoi cd
$EDITOR dot_zshrc.tmpl
```

Шаблоны в каталоге .chezmoitemplates должны создаваться вручную:

```
chezmoi cd
mkdir -p .chezmoitemplates
cd .chezmoitemplates
$EDITOR mytemplate
```

Редактирование файла шаблона

```
Используйте chezmoi edit:


chezmoi edit ~/.zshrc
```

Чтобы сделанные вами изменения сразу же применялись после выхода из редактора, используйте опцию –apply:

```
chezmoi edit --apply ~/.zshrc
```

Тестирование шаблонов

Тестирование с помощью команды chezmoi execute-template.

Тестирование небольших фрагментов шаблонов:

```
chezmoi execute-template '{{ .chezmoi.hostname }}'
```

Тестирование целых файлов:

```
chezmoi cd
chezmoi execute-template < dot_zshrc.tmpl
```

Синтаксис шаблона

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

4 Выполнение лабораторной работы

4.1 Менеджер паролей pass

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

Настройка

Ключи GPG

Просмотр списка ключей:

gpg –list-secret-keys

Если ключа нет, нужно создать новый:

gpg –full-generate-key

Инициализация хранилища

Инициализируем хранилище:

pass init

Синхронизация с git

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

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

Проверить статус синхронизации модно командой

pass git status

Настройка интерфейса с броузером

Для взаимодействия с броузером используется интерфейс native messaging. Поэтому кроме плагина к броузеру устанавливается программа, обеспечивающая интерфейс native messaging.

Плагин browserpass 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/we
ce/naepdomgkenhinolocfifgehidddafch.

Интерфейс для взаимодействия с броузером (native messaging) Репозиторий: https://github.com/browserpass/browserpass-native

Gentoo:

emerge www-plugins/browserpass

Fedora

dnf copr enable maximbaz/browserpass dnf install browserpass

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

Управление файлами конфигурации

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
grim
slurp

Установите шрифты:

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

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)”

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:

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

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

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

Ежедневные операции с chezmoi

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

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 вы довольны изменениями, вы можете применить их:

chezmoi apply

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, добавьте в файл конфигурации ~/.config/chezmoi/chezmoi.toml следующее:

[git] autoCommit = true autoPush = true

Всякий раз, когда в исходный каталог вносятся изменения, chezmoi фиксирует изменения с помощью автоматически сгенерированного сообщения фикса-

ции 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.

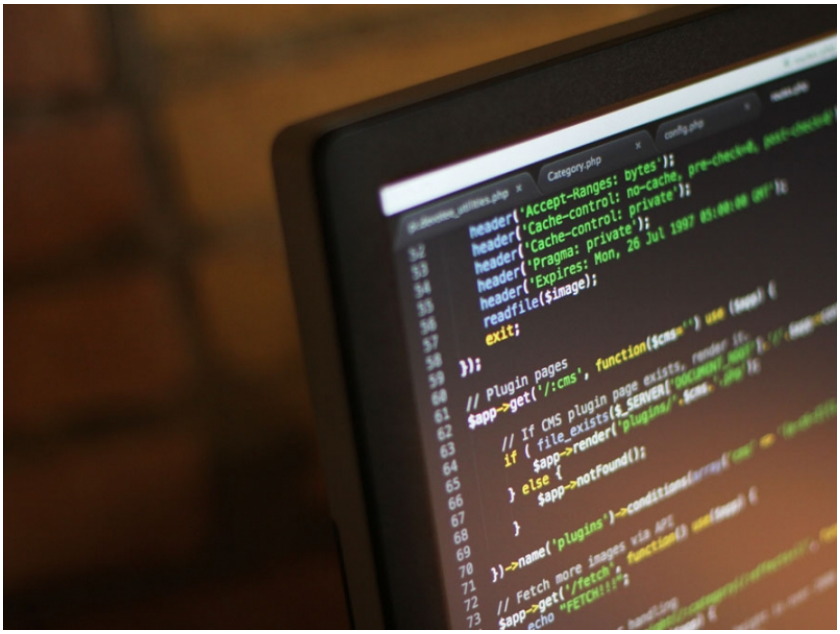

Рис. 4.1: Название рисунка

5 Выводы

Я научилась настраивать рабочую среду.