

Отчет по лабораторной работе №5

Операционные системы

Дмитрий Павлович Стрижов

Содержание

1	Цель работы	5
2	Задание	6
3	Теоретическое введение	7
4	Выполнение лабораторной работы	18
5	Выводы	23
	Список литературы	24

List of Figures

4.1	Установка менеджера паролей pass	18
4.2	Установка менеджера паролей pass	18
4.3	Список ключей	18
4.4	Инициализация хранилища	18
4.5	Создание структуры git	18
4.6	Задание адреса репозитория на хостинге	19
4.7	Синхронизация	19
4.8	Проверка статуса синхронизации	19
4.9	Установка плагина browserpass	20
4.10	Новый пароль	20
4.11	Установка дополнительного программного обеспечения	20
4.12	Подключение репозитория	20
4.13	Подключение репозитория	21
4.14	Установка шрифтов	21
4.15	Установка бинарного файла	21
4.16	Создание собственного репозитория	21
4.17	Инициализация и проверка последующих изменений	21
4.18	Внесение изменений	21
4.19	Извлечение из репозитория изменений	21
4.20	Подключение автоматического внесения изменений	22

List of Tables

1 Цель работы

Настройка рабочей среды.

2 Задание

1. Менеджер паролей pass
2. Управление файлами конфигурации

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.

Если же необходимо использовать дополнительное программное обеспечение, необходимо использовать 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 использоваться.

example.com.pgp

Соответствующее имя пользователя может быть именем файла внутри каталога, имя каталога может быть именем файла.

`example.com/user.pgp`

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

`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 соб

Реализации

Утилиты командной строки

На данный момент существует 2 основных реализации:

`pass` – классическая реализация в виде shell-скриптов (<https://www.passwordstore.org/>)

`gopass` – реализация на go с дополнительными интегрированными функциями (<https://www.gopass.in/>)

Дальше в тексте будет использоваться программа `pass`, но всё то же самое можно сде

Графические интерфейсы

`qtpass`

`qtpass` – может работать как графический интерфейс к `pass`, так и как самостоят

`gopass-ui`

`gopass-ui` – интерфейс к `gopass`.

`webpass`

Репозиторий: <https://github.com/emersion/webpass>

Веб-интерфейс к pass.

Написано на go lang.

Приложения для Android

Password Store

URL: <https://play.google.com/store/apps/details?id=dev.msfjarvis.aps>

Репозиторий с кодом: [https://github.com/android-password-store/Android-](https://github.com/android-password-store/Android-Password-Store)

Password-Store

Документация: <https://android-password-store.github.io/docs/>

Для синхронизации с git необходимо импортировать ssh-ключи.

Поддерживает разблокировку по биометрическим данным.

Для работы требует наличия OpenKeychain: Easy PGP.

OpenKeychain: Easy PGP

URL: <https://play.google.com/store/apps/details?id=org.sufficientlysecure.key>

Операции с ключами pgp.

Необходимо будет импортировать pgp-ключи.

Не поддерживает разблокировку по биометрическим данным. Необходимо набирать п

Пакеты для Emacs

pass

Основной режим для управления хранилищем и редактирования записей.

Emacs. Пакет pass

Репозиторий: <https://github.com/NicolasPetton/pass>

Позволяет редактировать базу данных паролей.

Запуск:

M-x pass

helm-pass

Интерфейс helm для pass.

Репозиторий: <https://github.com/emacs-helm/helm-pass>

Запуск:

M-x helm-pass

Выдаёт в минибуфере список записей из базы паролей. При нажатии Enter копирует

ivy-pass

Интерфейс ivy для pass.

Репозиторий: <https://github.com/ecraven/ivy-pass>

Управление файлами конфигурации

Использование chezmoi для управления файлами конфигурации домашнего каталога пользователя

Общая информация

Сайт: <https://www.chezmoi.io/>

Репозиторий: <https://github.com/twpayne/chezmoi>

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 и

Файлы, содержимое которых одинаково на всех ваших машинах, дословно копируются из

Файлы, которые варьируются от машины к машине, выполняются как шаблоны, обычно с

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 ваш репозиторий содержит файл с именем .chezmoi.\$FORMAT.tmpl, где \$FORMAT ес

Например, пусть ~/.local/share/chezmoi/.chezmoi.toml.tmpl выглядит так:

```
{{- $email := promptStringOnce . "email" "Email address" -}}
```

```
[data]
```

```
email = {{ $email | quote }}
```

При выполнении chezmoi init будет создан конфигурационный файл ~/.config/chez

promptStringOnce – это специальная функция, которая запрашивает у пользовател

Чтобы протестировать этот шаблон, используйте chezmoi execute-template с флагами init и --promptString, например:

```
chezmoi execute-template --init --promptString email=me@home.org < ~/.local/share
```

Пересоздание файл конфигурации

Если вы измените шаблон файла конфигурации, chezmoi предупредит вас, если ваш тек

Вы можете повторно сгенерировать файл конфигурации, запустив:

```
chezmoi init
```

Шаблоны

Общая информация

Шаблоны используются для изменения содержимого файла в зависимости от среды.

Используется синтаксис шаблонов Go.

Файл интерпретируется как шаблон, если выполняется одно из следующих условий:

- имя файла имеет суффикс `.tmpl`;

- файл находится в каталоге `.chezmoitemplates`.

Данные шаблона

Полный список переменных шаблона:

```
chezmoi data
```

Источники переменных:

- файлы `.chezmoi`, например, `.chezmoi.os`;

- файлы конфигурации `.chezmoidata.$FORMAT`. Форматы (`json`, `jsonc`, `toml`, `yaml`) чи

- раздел `data` конфигурационного файла.

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

```
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 `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.

Переменные записываются буквально:

```
{{ .chezmoi.hostname }}
```

Условные выражения могут быть записаны с использованием `if`, `else if`, `else`, `end`:

```
{{ if eq .chezmoi.os "darwin" }}
darwin
```

```
{{ else if eq .chezmoi.os "linux" }}
linux
```

```
{{ else }}
```

```
other operating system
```

```
{{ end }}
```

Удаление пробелов

Для удаления пробелов в шаблоне разместите знак минус и пробела рядом со скобками:

```
HOSTNAME={{- .chezmoi.hostname }}
```

В результате получим:

```
HOSTNAME=myhostname
```

Отладка шаблона

Используется подкоманда `execute-template`:

```
chezmoi execute-template '{{ .chezmoi.os }}/{{ .chezmoi.arch }}
```

Интерпретируются любые данные, поступающие со стандартного ввода или в конце файла.

Можно передать содержимое файла этой команде:

```
cat foo.txt | chezmoi execute-template
```

Логические операции

Возможно выполнение логических операций.

Если имя хоста машины равно work-laptop, текст между if и end будет включён в

```
# common config
export EDITOR=vi

# machine-specific configuration
{{- if eq .chezmoi.hostname "work-laptop" }}
# this will only be included in ~/.bashrc on work-laptop
{{- end }}
```

Логические функции

eq: возвращает true, если первый аргумент равен любому из остальных аргументов;
not: возвращает логическое отрицание своего единственного аргумента;
and: возвращает логическое И своих аргументов, может принимать несколько аргументов;
or: возвращает логическое ИЛИ своих аргументов, может принимать несколько аргументов.

Целочисленные функции

len: возвращает целочисленную длину своего аргумента;
eq: возвращает логическую истину arg1 == arg2;
ne: возвращает логическое значение arg1 != arg2;
lt: возвращает логическую истину arg1 < arg2;
le: возвращает логическую истину arg1 <= arg2;
gt: возвращает логическую истину arg1 > arg2;
ge: возвращает логическую истину arg1 >= arg2.

Переменные шаблона

Чтобы просмотреть переменные, доступные в вашей системе, выполните:

chezmoi data

Чтобы получить доступ к переменной `chezmoi.kernel.osrelease` в шаблоне, используйте

```
{{ .chezmoi.kernel.osrelease }}
```

4 Выполнение лабораторной работы

Устанавливаем необходимые программы (рис. 4.1, 4.2).

```
[dpstrizhov@dpstrizhov ~]$ sudo dnf install pass pass-otp  
[sudo] пароль для dpstrizhov:
```

Рис. 4.1: Установка менеджера паролей pass

```
[dpstrizhov@dpstrizhov ~]$ sudo dnf install gopass  
Последняя проверка окончания срока действия метаданных: 0:01:03 назад, Пн 11 мар 2024 01:25:07.  
Зависимости разрешены.
```

Рис. 4.2: Установка менеджера паролей pass

Проверяем список ключей (рис. 4.3).

```
Выполнено!  
[dpstrizhov@dpstrizhov ~]$ gpg --list-secret-keys
```

Рис. 4.3: Список ключей

Инициализируем хранилище (рис. 4.4).

```
[dpstrizhov@dpstrizhov ~]$ pass init 1132236054@pfur.ru  
mkdir: создан каталог '/home/dpstrizhov/.password-store/'  
Password store initialized for 1132236054@pfur.ru  
[dpstrizhov@dpstrizhov ~]$
```

Рис. 4.4: Инициализация хранилища

Создадим структуру git (рис. 4.5).

```
[dpstrizhov@dpstrizhov ~]$ pass git init  
Инициализирован пустой репозиторий Git в /home/dpstrizhov/.password-store/.git/  
[master (корневой коммит) ac37fb5] Add current contents of password store.  
1 file changed, 1 insertion(+)  
create mode 100644 .gpg-id  
[master 2f989a2] Configure git repository for gpg file diff.  
1 file changed, 1 insertion(+)  
create mode 100644 .gitattributes  
[dpstrizhov@dpstrizhov ~]$
```

Рис. 4.5: Создание структуры git

Задаем адрес репозитория на хостинге (рис. 4.6).

```
Переинициализирован существующий репозиторий Git в /home/dpstrizhov/.password-store/.git/  
[dpstrizhov@dpstrizhov ~]$ pass git remote add origin https://github.com/StrizhovDmitriy/haha.git
```

Рис. 4.6: Задание адреса репозитория на хостинге

Синхронизируем (рис. 4.7).

```
[dpstrizhov@dpstrizhov ~]$ pass git pull  
remote: Enumerating objects: 3, done.  
remote: Counting objects: 100% (3/3), done.  
remote: Total 3 (delta 0), reused 3 (delta 0), pack-reused 0  
Распаковка объектов: 100% (3/3), 856 байтов | 171.00 КиБ/с, готово.  
Из https://github.com/StrizhovDmitriy/haha  
* [новая ветка] master -> origin/master  
У текущей ветки нет информации об отслеживании.  
Пожалуйста, укажите с какой веткой вы хотите слить изменения.  
Для дополнительной информации, смотрите git-pull(1).  
  
git pull <внешний-репозиторий> <ветка>  
  
Если вы хотите указать информацию о отслеживаемой ветке, выполните:  
  
git branch --set-upstream-to=origin/<ветка> master  
  
[dpstrizhov@dpstrizhov ~]$ pass git push  
fatal: The current branch master has no upstream branch.  
To push the current branch and set the remote as upstream, use  
  
git push --set-upstream origin master  
  
To have this happen automatically for branches without a tracking  
upstream, see 'push.autoSetUpRemote' in 'git help config'.  
[dpstrizhov@dpstrizhov ~]$
```

Рис. 4.7: Синхронизация

Проверяем статус синхронизации (рис. 4.8).

```
[dpstrizhov@dpstrizhov .password-store]$ pass git status  
Текущая ветка: master  
Эта ветка соответствует «origin/master».  
  
ничего коммитить, нет изменений в рабочем каталоге  
[dpstrizhov@dpstrizhov .password-store]$
```

Рис. 4.8: Проверка статуса синхронизации

Устанавливаем плагин browserpass (рис. 4.9).

```
[dpstrizhov@dpstrizhov .password-store]$ sudo dnf copr enable maxibaz/browserpass
Включение репозитория Copr. Обратите внимание, что этот репозиторий
не является частью основного дистрибутива, и качество может отличаться.

Проект Fedora не имеет какого-либо влияния на содержимое этого
репозитория за рамками правил, описанных в Вопросах и Ответах Copr в
https://docs.pagure.org/copr.copr/user\_documentation.html#what-i-can-build-in-copr,
а качество и безопасность пакетов не поддерживаются на каком-либо уровне.

Не отправляйте сообщения об ошибках этих пакетов в Fedora
bugzilla. В случае возникновения проблем обращайтесь к владельцу этого репозитория.

Do you really want to enable copr.fedorainfracloud.org/maxibaz/browserpass? [y/N]: y
Ошибка: Не удалось активировать этот проект.
Проект «maxibaz/browserpass» не существует.
[dpstrizhov@dpstrizhov .password-store]$ sudo dnf copr enable maxibaz/browserpass
Включение репозитория Copr. Обратите внимание, что этот репозиторий
не является частью основного дистрибутива, и качество может отличаться.

Проект Fedora не имеет какого-либо влияния на содержимое этого
репозитория за рамками правил, описанных в Вопросах и Ответах Copr в
https://docs.pagure.org/copr.copr/user\_documentation.html#what-i-can-build-in-copr,
а качество и безопасность пакетов не поддерживаются на каком-либо уровне.

Не отправляйте сообщения об ошибках этих пакетов в Fedora
bugzilla. В случае возникновения проблем обращайтесь к владельцу этого репозитория.

Do you really want to enable copr.fedorainfracloud.org/maxibaz/browserpass? [y/N]: y
Репозиторий успешно подключен.
[dpstrizhov@dpstrizhov .password-store]$ sudo dnf install browserpass
Не найдена команда: install. Воспользуйтесь /usr/bin/dnf --help
Это, возможно, команда подключаемого модуля DNF, попробуйте: «dnf install 'dnf-command(install)'»
[dpstrizhov@dpstrizhov .password-store]$ sudo dnf install browserpass
```

Рис. 4.9: Установка плагина browserpass

Создаем новый пароль, а потом выводим его, а затем генерируем новый (рис. 4.10).

```
[dpstrizhov@dpstrizhov .password-store]$ pass insert password
An entry already exists for password. Overwrite it? [y/N] y
Enter password for password:
Retype password for password:
[master 32b3ab3] Add given password for password to store.
1 file changed, 0 insertions(+), 0 deletions(-)
[dpstrizhov@dpstrizhov .password-store]$ pass password
12345
[dpstrizhov@dpstrizhov .password-store]$ pass generate --in-place password
[master 64e0650] Replace generated password for password.
1 file changed, 0 insertions(+), 0 deletions(-)
The generated password for password is:
>472NMJJ*7B2;)ckt?l2kKJU5
[dpstrizhov@dpstrizhov .password-store]$ pass password
>472NMJJ*7B2;)ckt?l2kKJU5
[dpstrizhov@dpstrizhov .password-store]$
```

Рис. 4.10: Новый пароль

Устанавливаем дополнительное программное обеспечение (рис. 4.11).

```
[dpstrizhov@dpstrizhov ~]$ sudo dnf -y install \ dunst \ fontawesome-fonts \ powerline-fonts \ light \ fuzzel \
swaylock \ kitty \ waybar swaybg \ wl-clipboard \ mpv \ grim \ slurp
Последняя проверка окончания срока действия метаданных: 0:13:48 назад. Пн 11 мар 2024 01:59:59
```

Рис. 4.11: Установка дополнительного программного обеспечения

Устанавливаем шрифты (рис. 4.12, 4.13, 4.14).

```
[dpstrizhov@dpstrizhov ~]$ sudo dnf copr enable peterwu/iosevka
Включение репозитория Copr. Обратите внимание, что этот репозиторий
не является частью основного дистрибутива, и качество может отличаться.
```

Рис. 4.12: Подключение репозитория

```
[dpstrizhov@dpstrizhov ~]$ sudo dnf search iosevka
Copr repo for iosevka owned by peterwu                               31 kB/s | 53 kB    00:01
Последняя проверка окончания срока действия метаданных: 0:00:01 назад, Пн 11 мар 2024 02:19:51.
```

Рис. 4.13: Подключение репозитория

```
[dpstrizhov@dpstrizhov ~]$ sudo dnf install iosevka-fonts iosevka-aile-fonts iosevka-curly-fonts iosevka-slab-fonts iosevka-etoile-fonts iosevka-term-fonts
[sudo] пароль для dpstrizhov:
```

Рис. 4.14: Установка шрифтов

Устанавливаем бинарный файл (рис. 4.15).

```
Выполнено!
[dpstrizhov@dpstrizhov ~]$ sh -c "$(wget -qO- chezmoi.io/get)"
```

Рис. 4.15: Установка бинарного файла

Создаем собственный репозиторий с помощью утилит (рис. 4.16).

```
info installed ./bin/chezmoi
[dpstrizhov@dpstrizhov ~]$ gh repo create dotfiles --template="yamadhazma/dotfiles-tempale" --private
```

Рис. 4.16: Создание собственного репозитория

Проводим инициализацию и проверяем изменения (рис. 4.17).

```
git-extended  README.md  Видео  Загрузки  Музыка  Рабочий стол
dpstrizhov@dpstrizhov ~]$ chezmoi init https://github.com/StrizhovDmitriy/dotfiles.git
dpstrizhov@dpstrizhov ~]$ chezmoi diff
```

Рис. 4.17: Инициализация и проверка последующих изменений

Вносим изменения (рис. 4.18).

```
[dpstrizhov@dpstrizhov ~]$ chezmoi apply -v
```

Рис. 4.18: Внесение изменений

Извлекаем из репозитория изменения (рис. 4.19).

```
dpstrizhov@dpstrizhov ~]$ chezmoi update
уже актуально.
dpstrizhov@dpstrizhov ~]$ mc

dpstrizhov@dpstrizhov ~$ chezmoi.j$ sudo nano chezmoi.toml
sudo/ пароль для dpstrizhov:
dpstrizhov@dpstrizhov ~$ chezmoi.j$
```

Рис. 4.19: Извлечение из репозитория изменений

Подключение автоматического внедрения изменений (рис. 4.20).

```
[data]  
email = "1132230854@prtg.ru"  
[git]  
autoCommit = true  
autoPush = true
```

Рис. 4.20: Подключение автоматического внедрения изменений

5 Выводы

За время выполнения лабораторной работы я настроил рабочую систему.

Список литературы

Лабораторная работа №5: <https://esystem.rudn.ru/mod/page/view.php?id=1098796>