Functional package management in a pill

- Dolstra, Eelco (2006). "The Purely Functional Software Deployment Model" (Ph.D.). Utrecht University
- pioneered by Nix
- also employed by GNU Guix
- no Filesystem Hierarchy Standard (no /usr/bin, /usr/share, etc.)
- packages live in a **store** directory, e.g.
  - /gnu/store/y0d8ab1mi6lh0a3vpn5ly4l4sxq9wbn4x-orc-0.4.32
  - /gnu/store/9pypr3c3y379shhbwm9ilb4ipk9mkfdd3-mesa-22.2.4
  - /gnu/store/rv91v4x30kcjh7xq6k4l2njikk79frxk-freeglut-3.4.0
  - /gnu/store/30sfbjqxrsk2wg8nhxd1xjx3q3n976fsz-lesa-608
- a daemon builds packages from definitions and puts them in the store

- we're using GNU Guix here (no, not the trademarked GUIX...)

- store filename determine by hash of package inputs + definition

- multiple versions of a package can coexist

- per-project development environments

- easy rollbacks

- emphasis on reproducible builds

Functional package management in a pill (sample package)

```
$ cd /gnu/store/30zfbjazrzk2wg8nhsd1sgi3q3s9796x-less-608/
$ find . -type f
./bin/less
./bin/lessecho
./bin/lesskey
./etc/ld.so.cache
./share/doc/less-608/LICENSE
./share/doc/less-608/COPYING
./share/man/man1/lessecho.1.gz
./share/man/man1/lesskey.1.gz
./share/man/man1/less.1.gz
$ ls -lh bin/less
-r-xr-xr-x 2 root root 192K Jan  1  1970 bin/less
```

- store is read-only (only Nix/Guix daemon can write)

- store files are root-owned and world-readable =¿ secrets must be managed differently

- dates set to Epoch (but ls -lch shows real creation time)

- the same package won't be built twice, even if requested by multiple users

- a package will built again (or grafted) when one of its dependencies gets updated

- a package not in use can be garbage-collected

- no support for quotas yet as of 2024

CTF — GNU Guix storefile mistake

└─Functional package management in a pill (declarative OS)

Functional package management in a pill (declarative OS)

- **packages** are defined declaratively
- **services** are defined declaratively as well
- service **configurations** are defined declaratively *as well*. . .
  ```
  (service httpd-service-type
          (httpd-configuration
            (config
              (httpd-config-file
                (server-name "www.example.com")
                (document-root "/var/public_html")))))
  ```
  - . . .and result in store files like
    /gnu/store/54vxa5x1b75simbvuhsqkfxsjk040al-httpd.conf
- Yay, we can replace Ansible! But what about secrets?
  - option 1: keep private keys and passwords outside the store
  - option 2: put them encrypted in the store

- GNU Guix and Nix have their DSLs (the first one is actually Scheme Lisp + some APIs)

- on Guix/Nix server packages and configurations are immutable (we can switch to different ones but not alter the existing ones) — convenient

- an application may require database credentials, some API token, a private key for TLS certificate, etc.

- encrypted secrets in store — one master key kept outside the store

CTF — GNU Guix storefile mistake

└─Sensitive information exposure scenario

Sensitive information exposure scenario

challenge — password hunt in /gnu/store

"You're an employee of a secret government agency. Analysis of
wiretap recordings have lead the agency to believe that an
individual known as Abdul Al-Inh-Ohn-Ih has come into possession
of highly classified government documents. If this turn out true
and Abdul blows the whistle on information from those materials,
years of intelligence efforts shall be ruined.

Abdul has been using the Matrix protocol for some of his
communication. Your current task is to get access to his Matrix
account. Start your investigation by taking a look at his blog."

A user of certain shared GNU Guix system has put a secret (a password) in
/gnu/store by mistake. The CTF competitioneer has to SSH into another
account on said system and find the password.

- we have some lore

- real-world references might be intended or not...

- no direct info about the exposures (one needs to figure this out)

Investigation (Abdul's blog)

Abdul's technology blog

Hi, I'm Abdul El-the-Oh-n-th. I'm a enthusiast in programming. Lisp is mine favourite language. I also essay to write it. I write about stuff I do on our computer(s).

Below is my index of mine blog entries. If something there is interesting you, feel free emailing me at abdul2007@pm.me.

Note: some of articles are available only in gemini:// Use an gemini browser for view that.

https://eeensrootout.net/

https://eeensrootout.net/software/

Mine entries

2024-03-15

Matterbridge in Ridgearval

2024-02-28

- language — itself a hint Abdul is likely to make mistakes

- only the few relevant blog entries (no misleading of competitioneers)

- mechanics of Guix relevant to the challenge are touched in the posts

- some extra effort required — obtaining a Gemini browser

CTF — GNU Guix storefile mistake

└─Investigation (peeking through Gemini)

Investigation (peeking through Gemini)

- most relevant parts of blog only accessible through Gemini (a lightweight alternative to HTTP)

- a Gemini browser "Lagrage" recommended in HTTP part of Abdul's blog

Investigation (Spotting mistakes)

configuration which hits a mistake is included in Abdul's blog

```
;;; ...
(list (shepherd-service
        (provision '(mattermost))
        (modules '((shepherd support)))     ;for '%user-log-dir'
        (start #~(make-forkexec-constructor
                  (#$(file-append matterbridge "/bin/matterbridge")
                   "--conf"
                   #$(local-file "config.toml"))
                  #:log-file (string-append %user-log-dir
                                            "/matterbridge.log")))
        (stop #~(make-kill-destructor))
        (documentation "Start local matterbridge.")))))
;;; ...
```
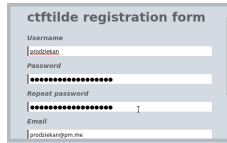
- the config suggests Matrix password is in config.toml in /gnu/store

Investigation (Account creation)

ctftilde registration form

Username
prodziekan

Password
●●●●●●●●●●●●●●●

Repeat password
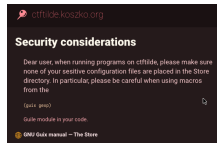●●●●●●●●●●●●●●●

Email
prodziekan@pm.me

- both Abdul's blog and the server's main website urge one to make an account and log in to the tilde server with SSH

- emails entered not actually used

CTF — GNU Guix storefile mistake

└─Hint 1

- page with the hint accessible through Gemini only

CTF — GNU Guix storefile mistake

└─Hint 2

Scheme Procedure: local-file *file* [*name*]
[#:recursive? #t] [#:select? (const #t)]

Return an object representing local file *file* to add to the store; this object can be used in a gexp. If *file* is a literal string denoting a relative file name, it is looked up relative to the source file where it appears; if *file* is not a literal string, it is looked up relative to the current working directory at run time. *file* will be added to the store under *name*—by default the base name of …

- link to GNU Guix HTML documentation

- suggestion that it has sth to do with the local-file macro (used in Abdul's code)

Finding the flag

```
~$ (cd /gnu/store && ls -cht *config.toml*)
qmdh299prl1p4fygw893w001v9ypi5z2-config.toml
~$

rather expected contents of
qmdh299prl1p4fygw893w001v9ypi5z2-config.toml

# ...
[matrix.ncevil-pl]
Server="https://matrix.ncevil.pl"
Login="abdul"
Password="fl&\u0067{full_source-bootstrap}"
RemoteNickFormat="[{PROTOCOL}] <{NICK}> "
NoHomeServerSuffix=false
# ...
```

- "g" in flag replaced with unicode escape to make bypassing with
  recursive grepping harder