Tarides

HOMOGENEOUS BUILDS WITH OCAML

OBUILDER

INTRODUCTION

- a lightweight sandboxing solution OBuilder
- support for linux, macOS, Windows and the BSDs
- avoid expensive virtualisation
- in production providing infrastructure for opam-health-check, opam-repo-ci and ocaml-ci

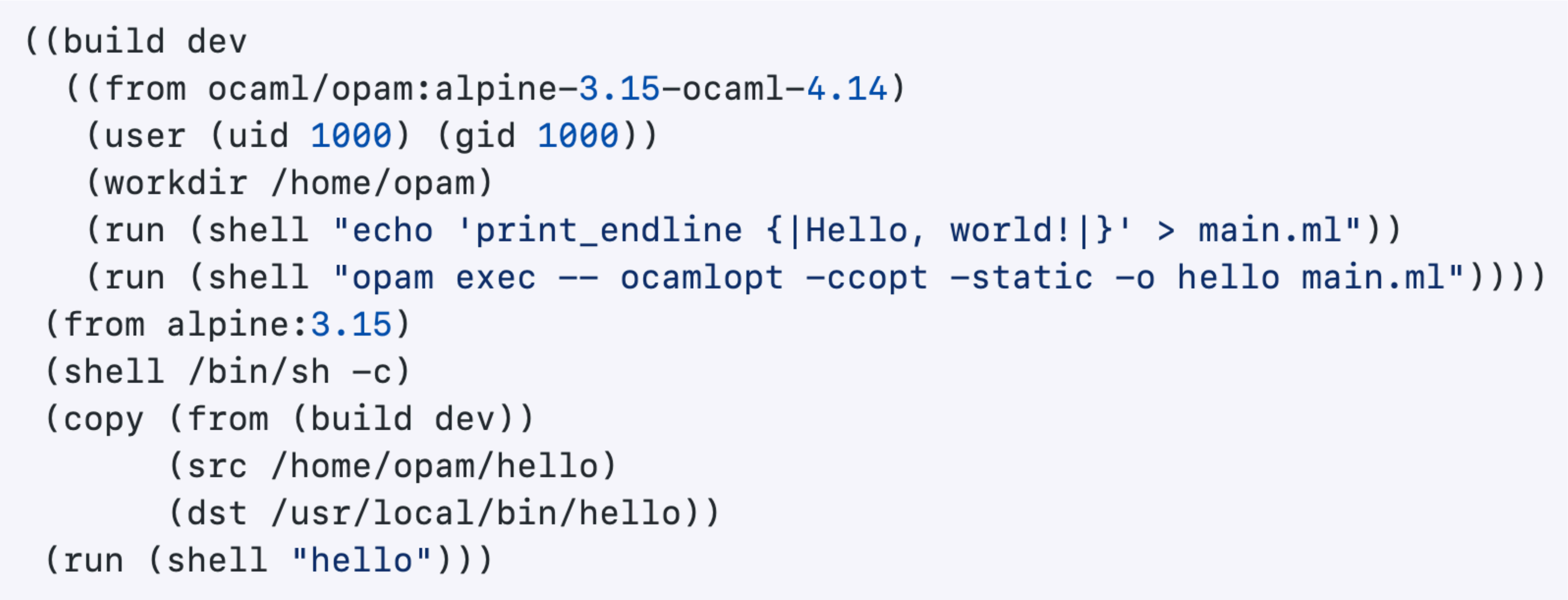
BACKGROUND

Our requirements:

- run OCaml on many operating systems and architectures
- various continuous integration and build systems
- common build specification
- support for Linux, macOS, Windows and BSDs
- multi-architecture (x86, ARM64, PPC64, s390x, RISC-V)

BUILD SPECIFICATIONS

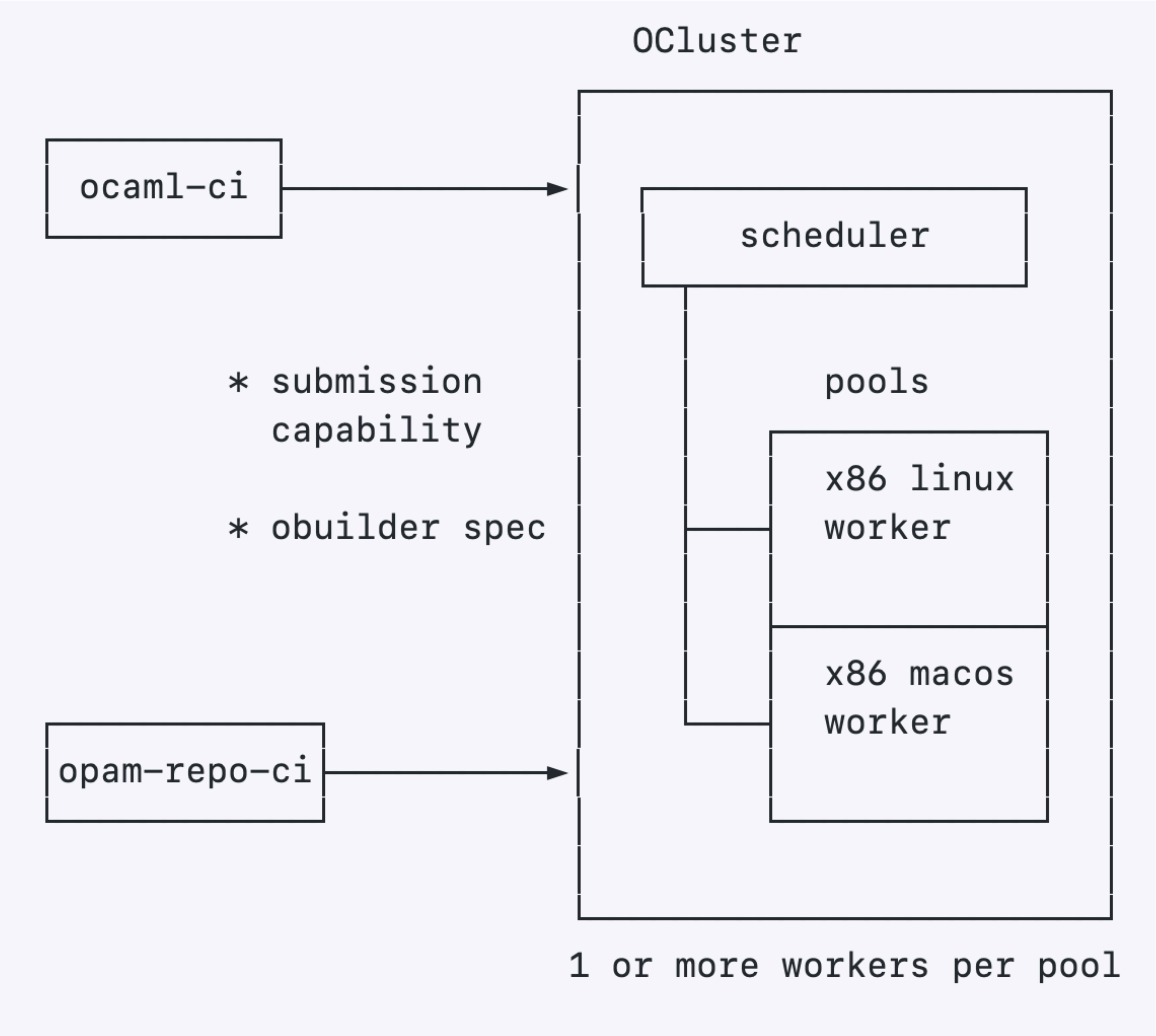
OBuilder is designed to take a build script and perform the steps in it inside a sandboxed environment.



```
((build dev
  ((from ocaml/opam:alpine-3.15-ocaml-4.14)
   (user (uid 1000) (gid 1000))
   (workdir /home/opam)
   (run (shell "echo 'print_endline {|Hello, world!|}' > main.ml"))
   (run (shell "opam exec -- ocamlopt -ccopt -static -o hello main.ml"))))
 (from alpine:3.15)
 (shell /bin/sh -c)
 (copy (from (build dev))
       (src /home/opam/hello)
       (dst /usr/local/bin/hello))
 (run (shell "hello")))
```

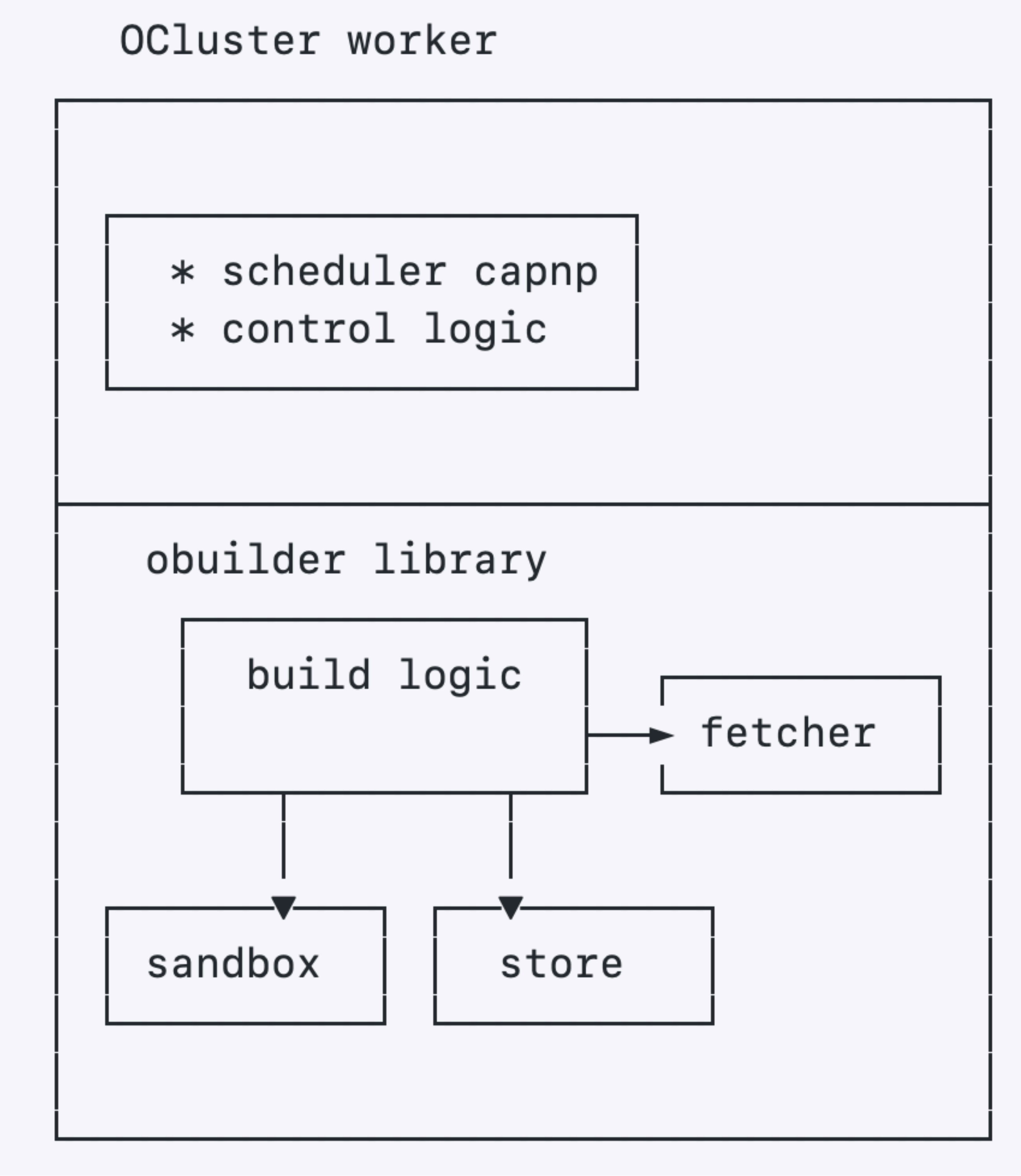
OCLUSTER ARCHITECTURE

- various clients require builds
- OCluster providing a build cluster of pools
- Pools as workers with similar capabilities
- Workers execute the build spec



OBUILDER ARCHITECTURE

- responsible for executing a build spec
- coordinates communication with the cluster
- provides a sandbox, store and fetcher
- workers are platform specific
- OBuilder used as a library



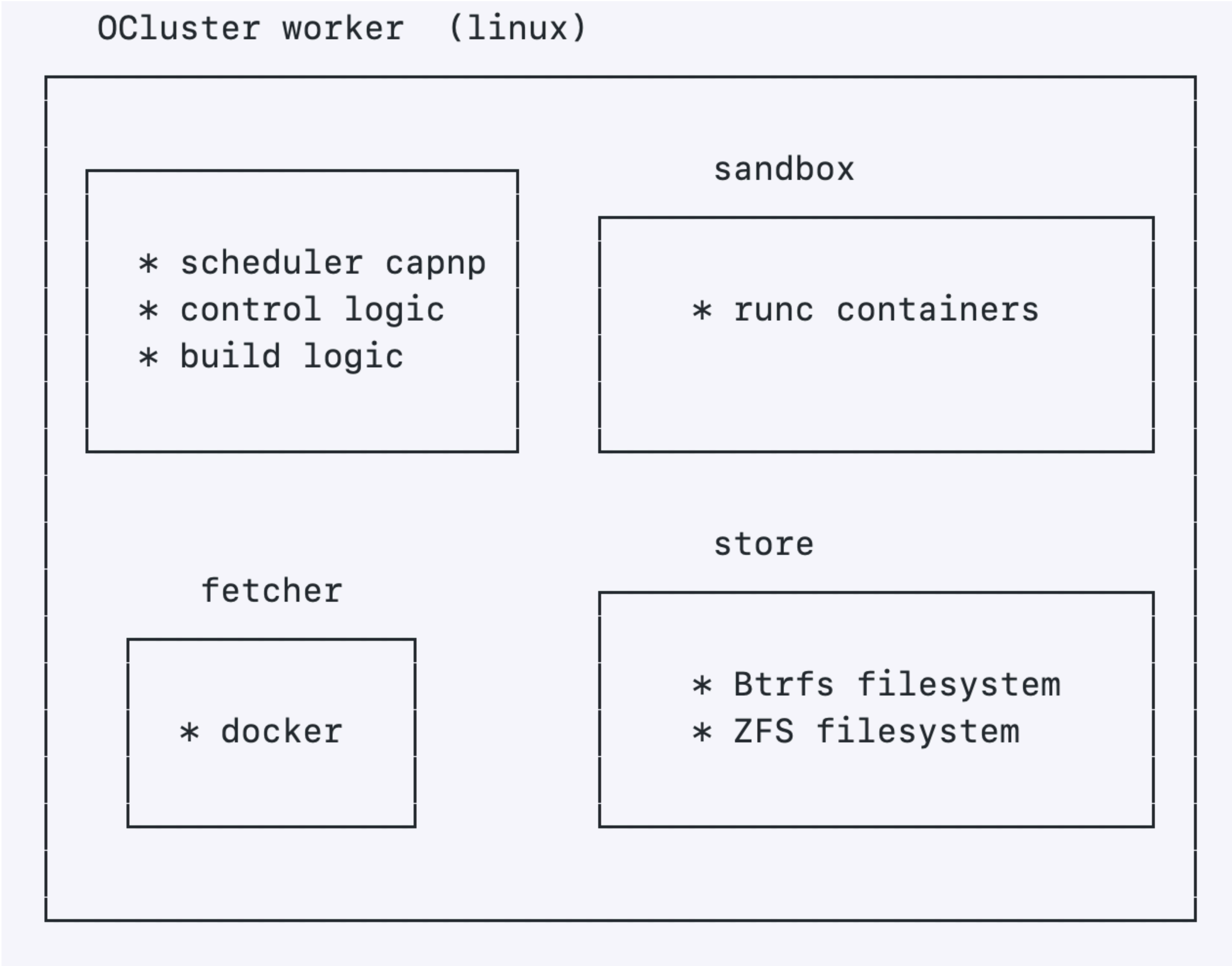
LINUX IMPLEMENTATION

Sandbox:

- provided by runc and native containerisation
- based on Linux's native namespaces and cgroups functionality

Store:

- Btrfs store uses Btrfs subvolumes to snapshot and restore filesystem state
- ZFS store uses the native snapshotting support in ZFS



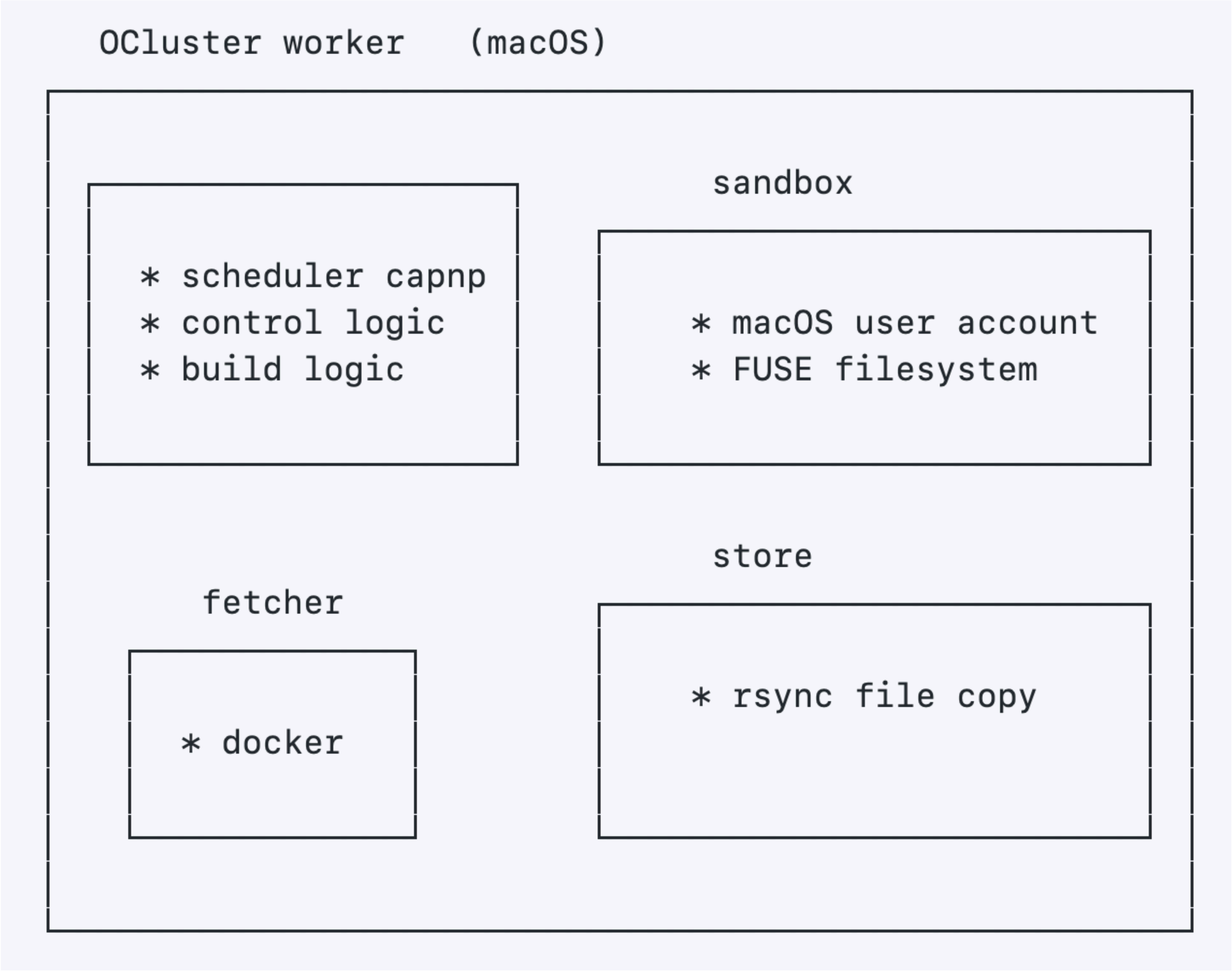
MACOS IMPLEMENTATION

Sandbox:

- provided by user accounts
- complications with opam and homebrew
- opam supports multiple opam roots on a system
- homebrew tricked by file system redirection, per user homebrew install

Store:

- provided by rsync and file copying
- rejected option of ZFS and FUSE



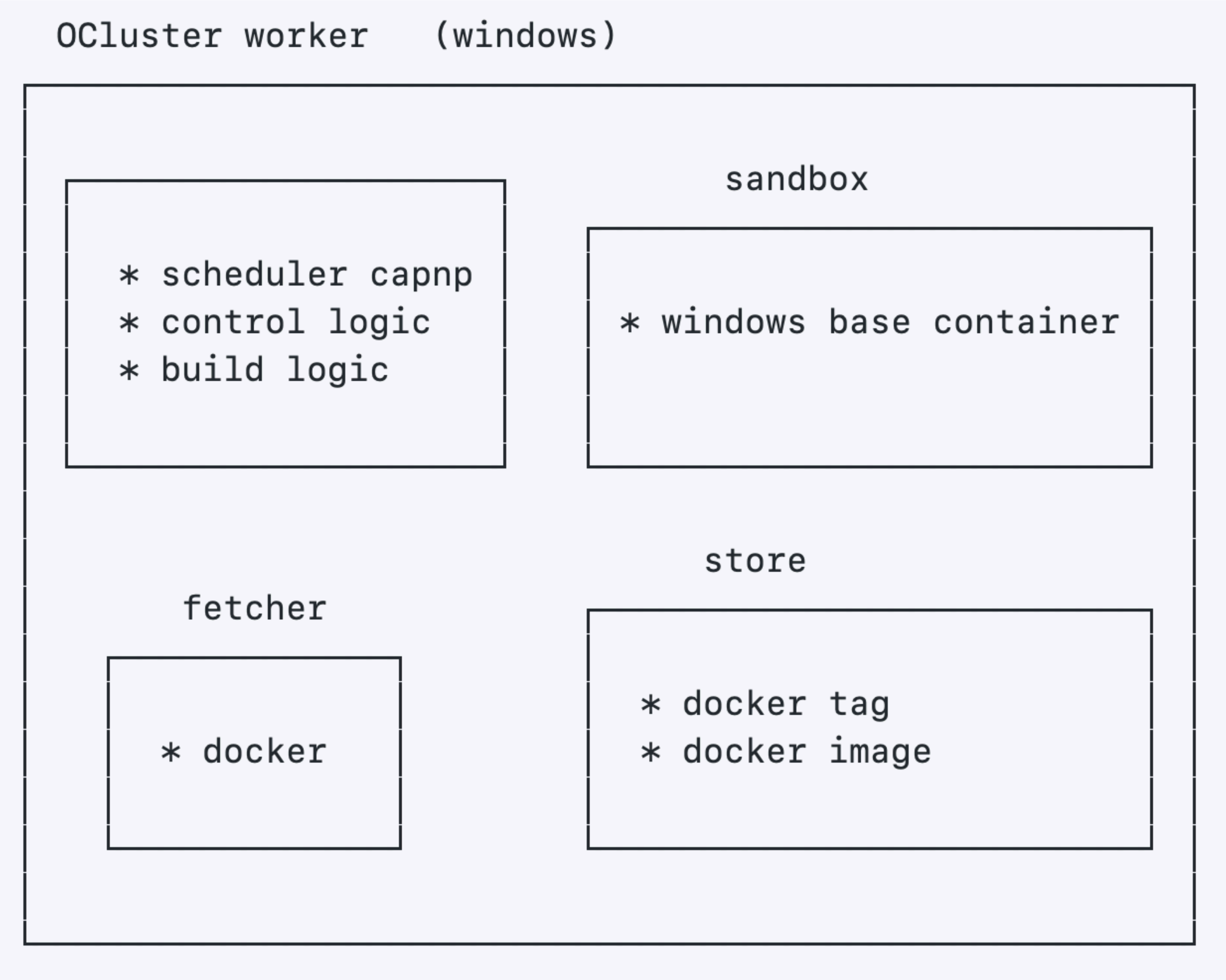
WINDOWS IMPLEMENTATION

Sandbox:

- uses a Windows docker container with opam and ocaml
- runs the build spec inside the container

Store:

- docker tag the running container
- store the resulting image under the computed key for that build step



FUTURE WORK

- MacOS support for ocaml-ci
- MacOS scaling to multi-user and ZFS stability
- Windows support for opam-repo-ci then ocaml-ci
- Revisiting native Windows solutions
- FreeBSD support using runj and ZFS

THANKS

- Longer PDF/LaTex version with further details https://github.com/tmcgilchrist/ocaml-2022-submission
- https://github.com/ocurrent/obuilder
- https://github.com/ocurrent/ocluster