containerd port to darwin

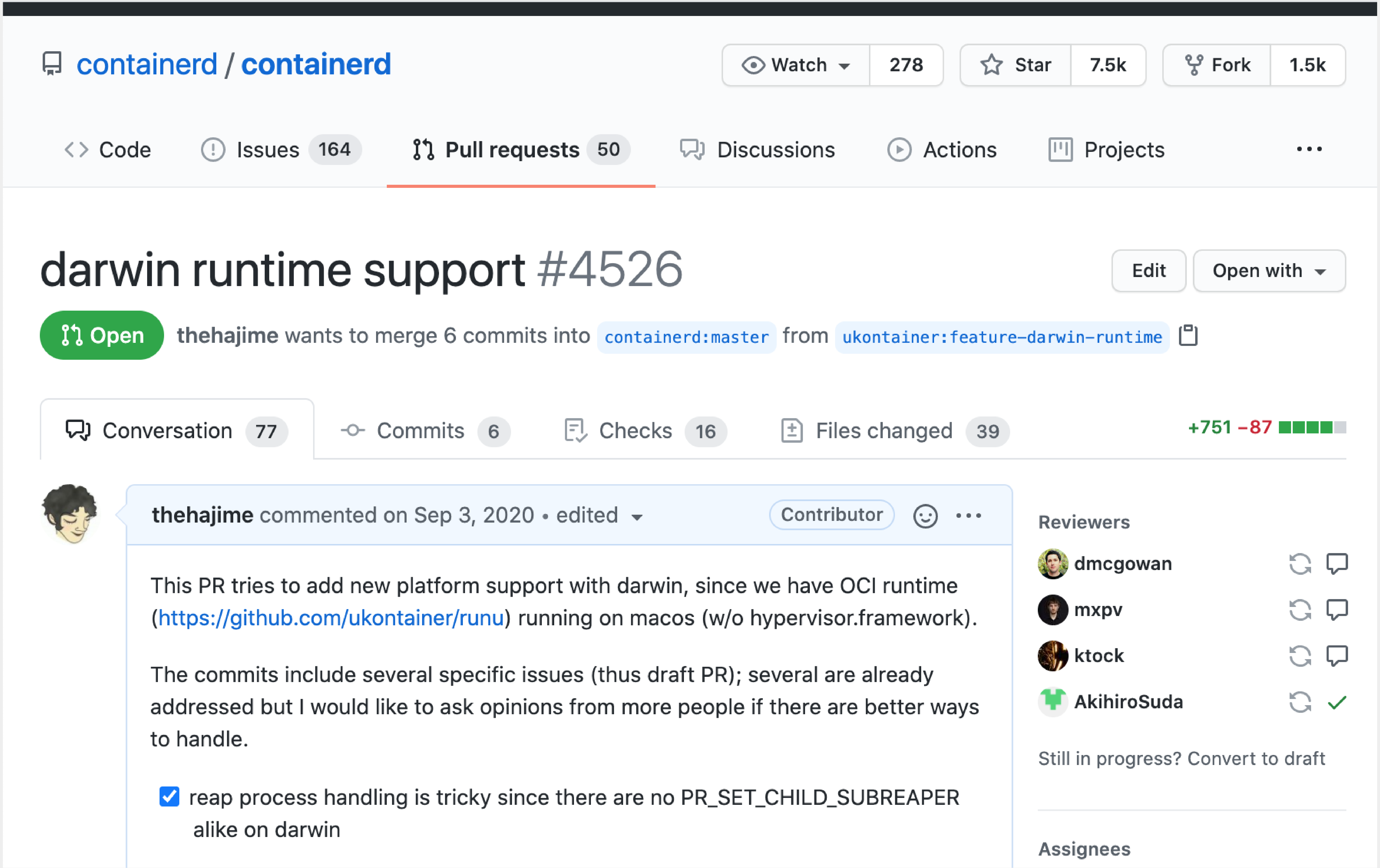
Toward Running Linux containers on macOS

Hajime Tazaki (@thehajime)

IIJ Research Laboratory

FOSDEM 2021: February 2021

Room: D.containers



darwin runtime support #4526

Open **thehajime** wants to merge 6 commits into `containerd:master` from `ukontainer:feature-darwin-runtime`

thehajime commented on Sep 3, 2020 • edited

This PR tries to add new platform support with darwin, since we have OCI runtime (https://github.com/ukontainer/runu) running on macos (w/o hypervisor.framework).

The commits include several specific issues (thus draft PR); several are already addressed but I would like to ask opinions from more people if there are better ways to handle.

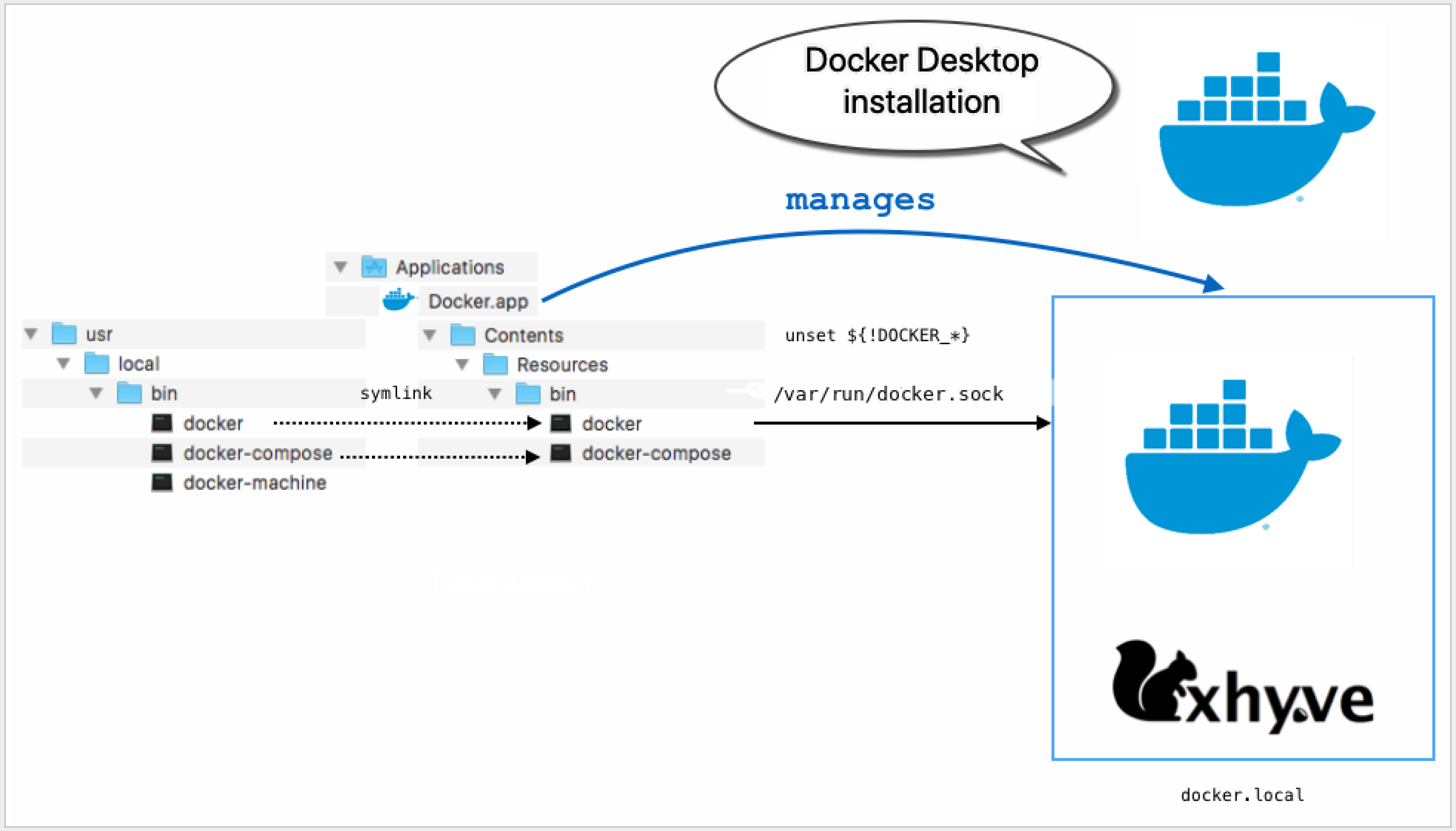
☑ reap process handling is tricky since there are no PR_SET_CHILD_SUBREAPER alike on darwin

- Pull request: darwin runtime support (containerd)

Docker on macOS (Docker Desktop)

- Run Linux programs (container) on foreign platform (Windows/macOS)
  - Small Linux VM
  - everything (e.g., containerd) runs on VM
- Goal: Transparent usage of Linux containers
- Useful for development environment

You don't really need containerd for darwin platform



ref: https://docs.docker.com/docker-for-mac/images/docker-for-mac-install.png

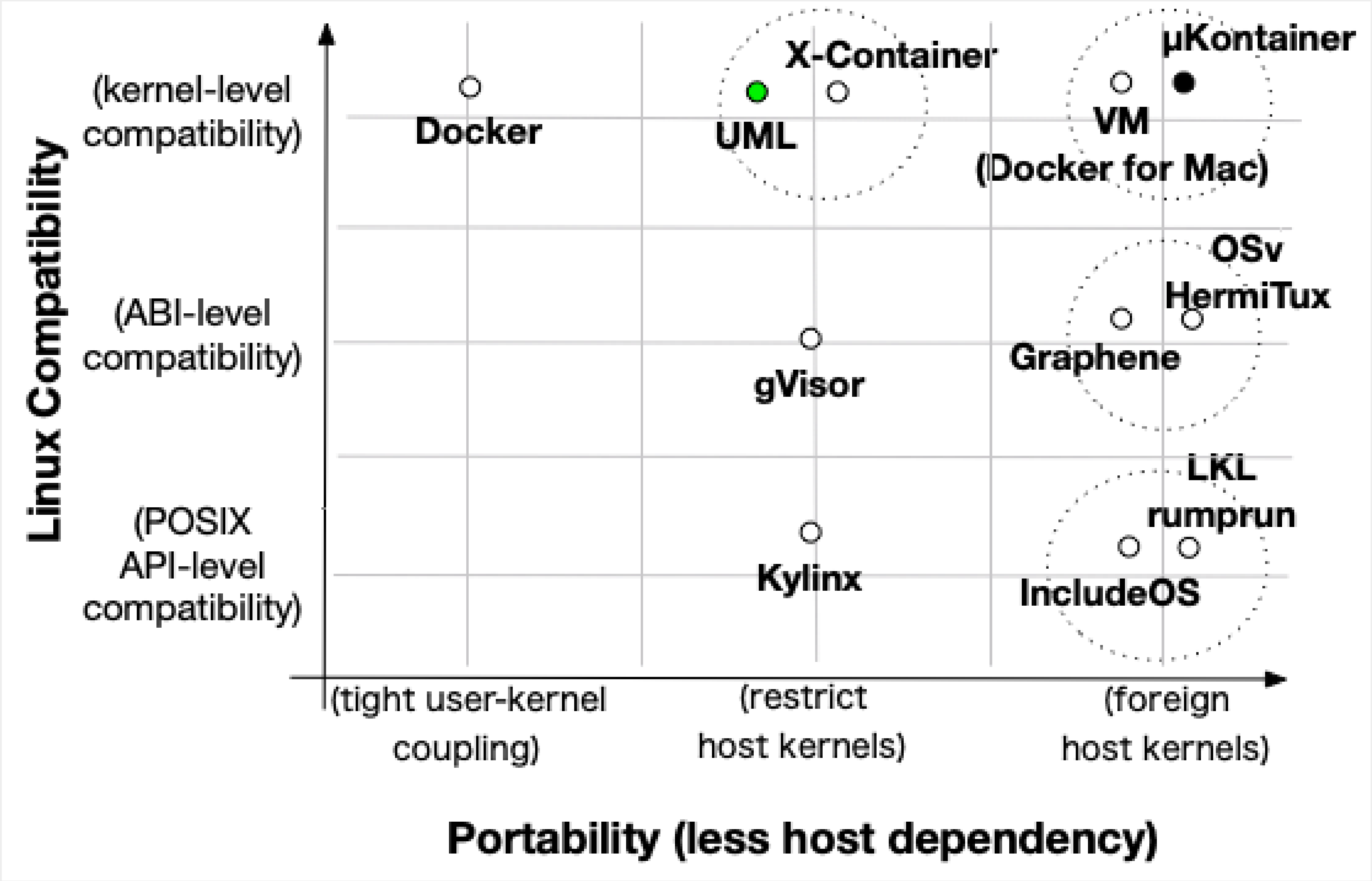
Motivations

Running Linux applications on macOS

- Linux kernel-like emulation projects
  - WSL (Windows Subsystem for Linux)
  - Graphene
  - Noah
  - gVisor
- Lightweight Linux virtualization on macOS
  - Docker Desktop
  - OSv
  - Firecracker?
  - hyper.sh (kata containers)

image: https://linuxnewbieguide.org/how-to-install-linux-on-a-macintosh-computer/

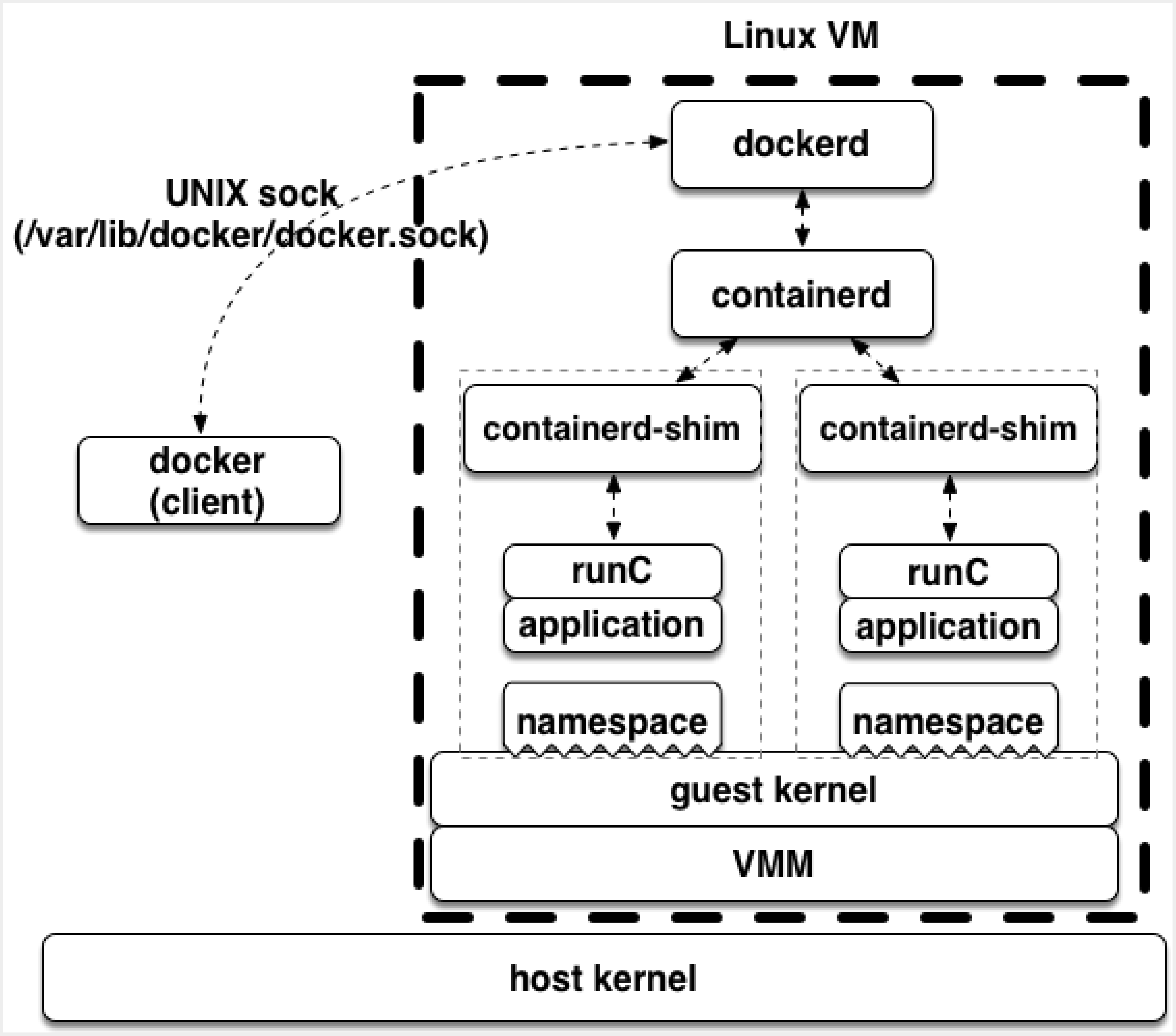
Motivations (cont'd)



- Running VMs still requires heavy-lifting
- Running Linux emulators tend to be incomplete
  - **We don't wish to re-write Linux kernel**

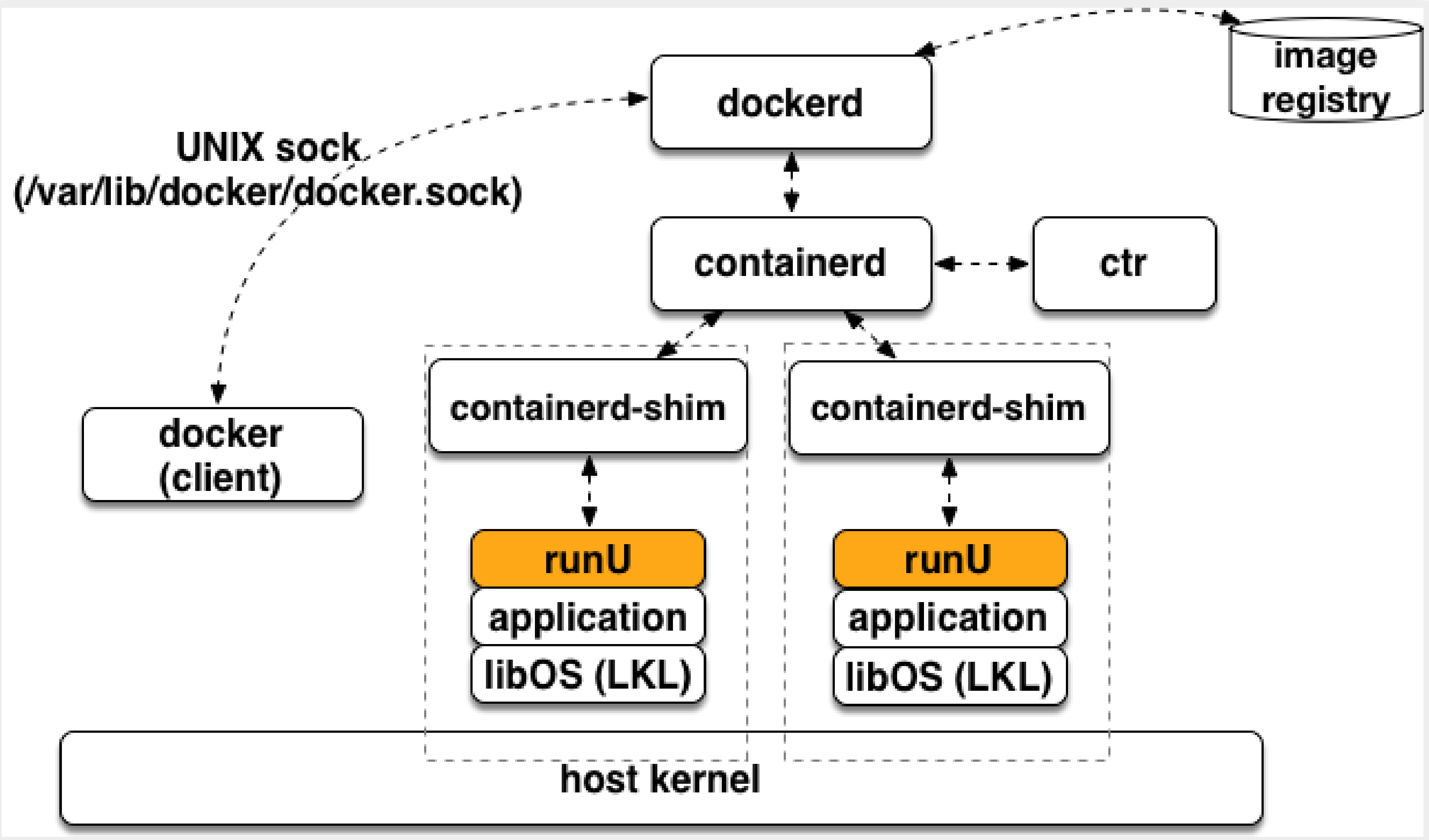
Goal: VM-level compatibility while Container-level lightweight property

Internals: Docker macOS



- containerd, dockerd, runc, applications run on Linux VM
- What's missing ?
  - no dockerd for darwin
  - no containerd for darwin
  - no OCI runtime (runc, etc) for darwin

Internals: Docker macOS++

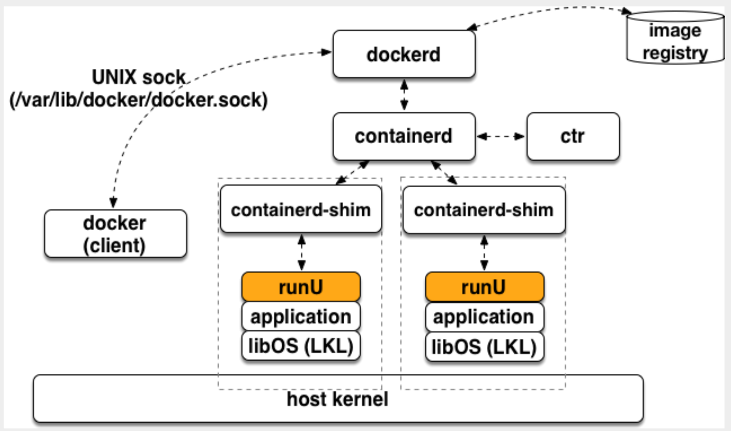


- Components
  - containerd (darwin)
  - dockerd (darwin)
  - OCI runtime: runu
  - **library OS** (LKL)

- Run docker images **without** Hypervisor.framework
  - as Mach-O (user space) programs
- Programs except container image are Mach-O binaries
  - Benefits native experience while doing Linux
- Currently only x86_64 works (both mac and container image)
  - effort to Apple Silicon support is ongoing

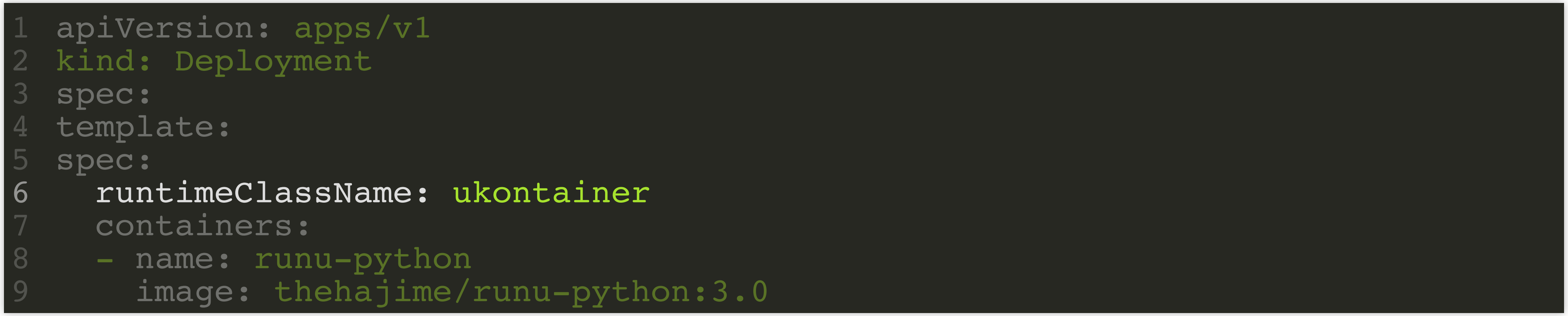
containerd: darwin port

- containerd-shim: already available (for what?)
- **only port runtime-independent implementation**
  - runu is not only the OCI runtime
- snapshotter: use native (add a bit of missing stubs)
- adapt darwin/XNU behavior as *ifdefs*
  - mount operation (no bind mount => symlink)
  - different syscall behaviors (fchown, etc)
  - different fork/subreaper behavior
  - eliminate missing Linux features (cgouprs, oom, etc)
- add macOS CI instance (tests)



OCI runtime: runu

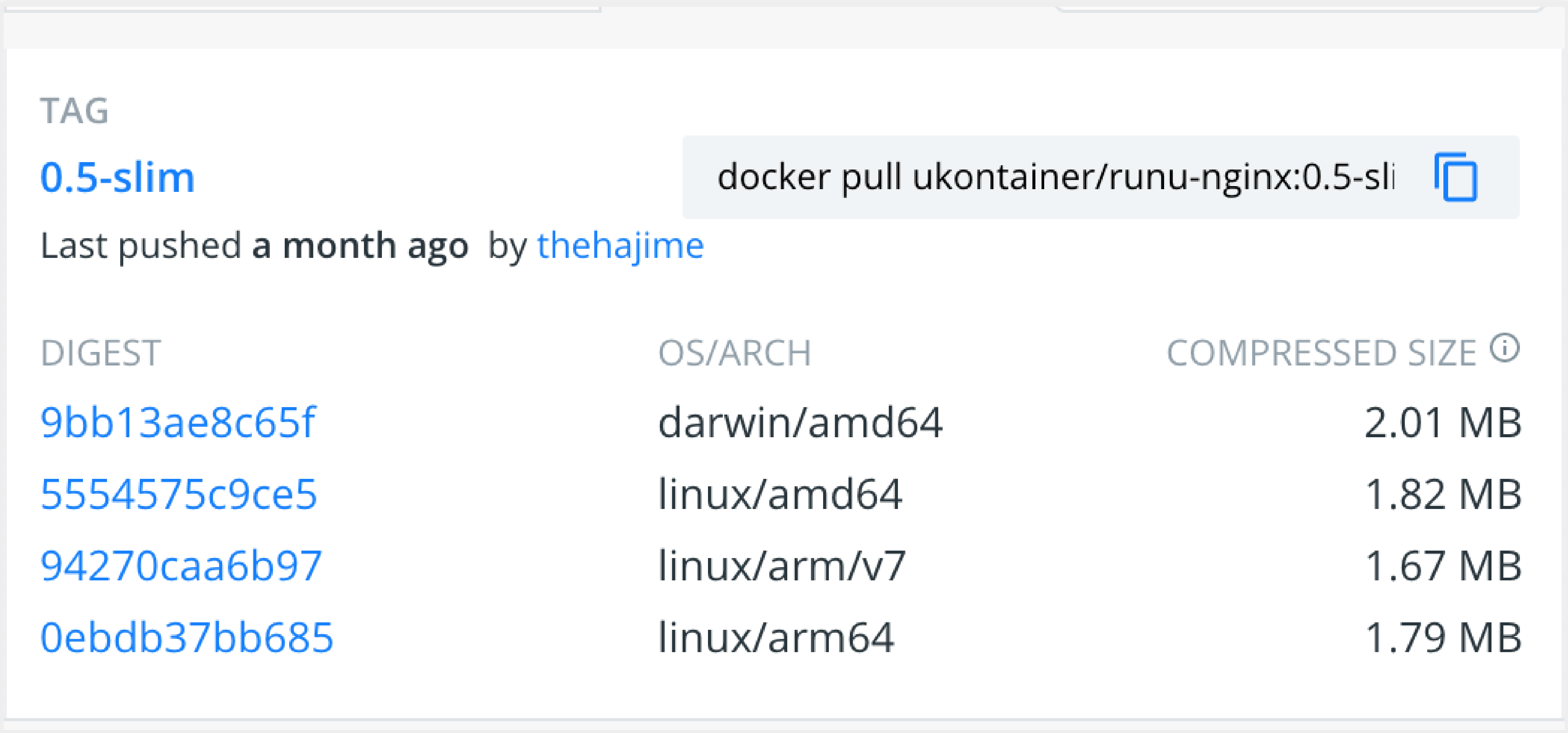
- Run LKL (Linux Kernel Library) programs under docker/k8s
- Communicate w/ containerd/kubelet
  - setup (virtual) devices as exposed file descriptors (fds)
  - (tap, veth, disk image, virtio 9pfs)
  - (optionally) replace libc.so
- Images
  - runu-private image (statically-linked LKL application)
  - public image (e.g., `alpine:latest`) (libc replacement)
- usage
  - Docker: `docker run` ***`--runtime=runu`*** `runu-python:latest`
  - k8s: add a `runtimeClassName` line



```yaml
apiVersion: apps/v1
kind: Deployment
spec:
template:
spec:
  runtimeClassName: ukontainer
  containers:
  - name: runu-python
    image: thehajime/runu-python:3.0
```

OCI runtime (cont'd)

- Multi-arch images



TAG

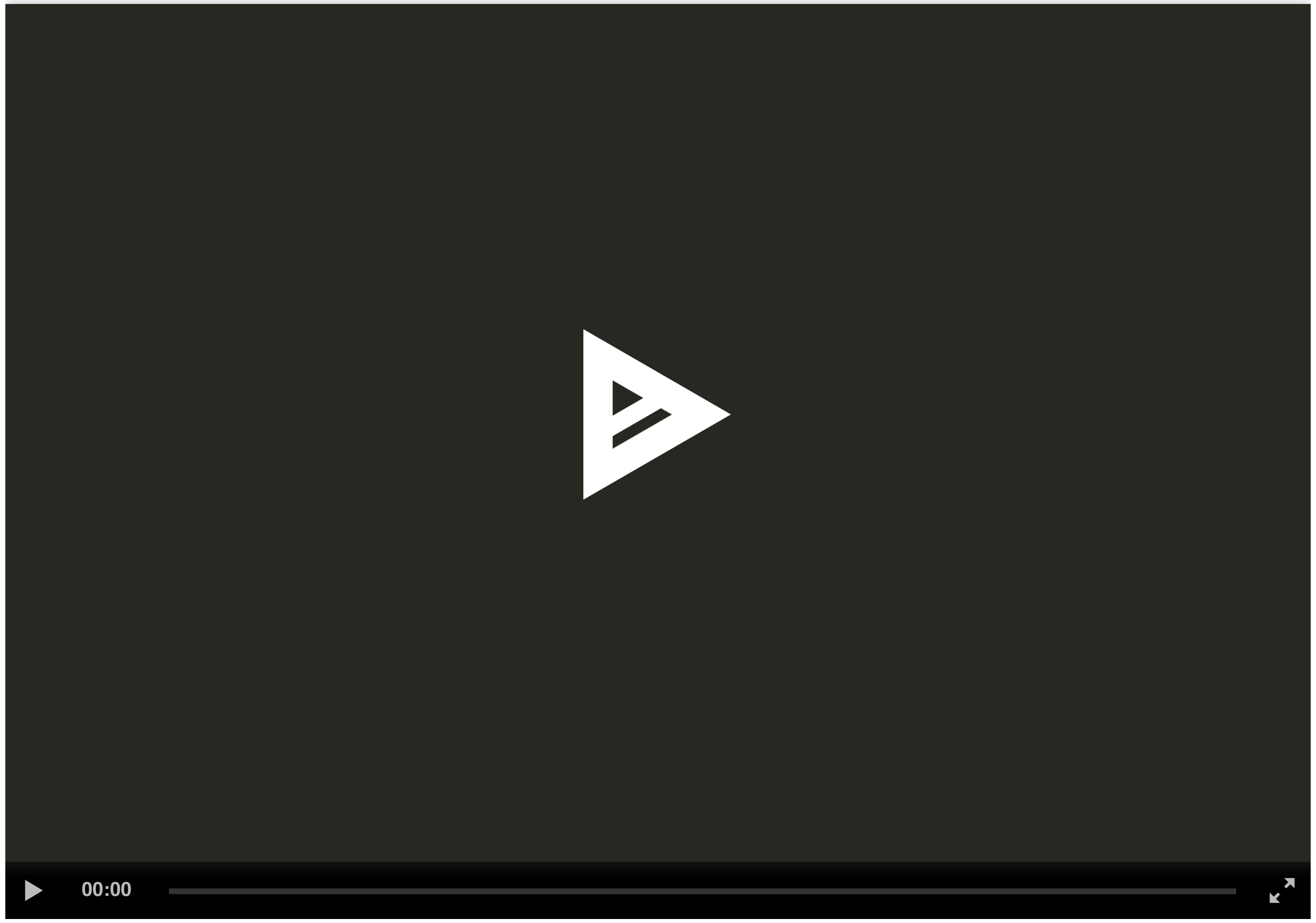
0.5-slim

```
docker pull ukontainer/runu-nginx:0.5-sli
```

Last pushed **a month ago** by thehajime

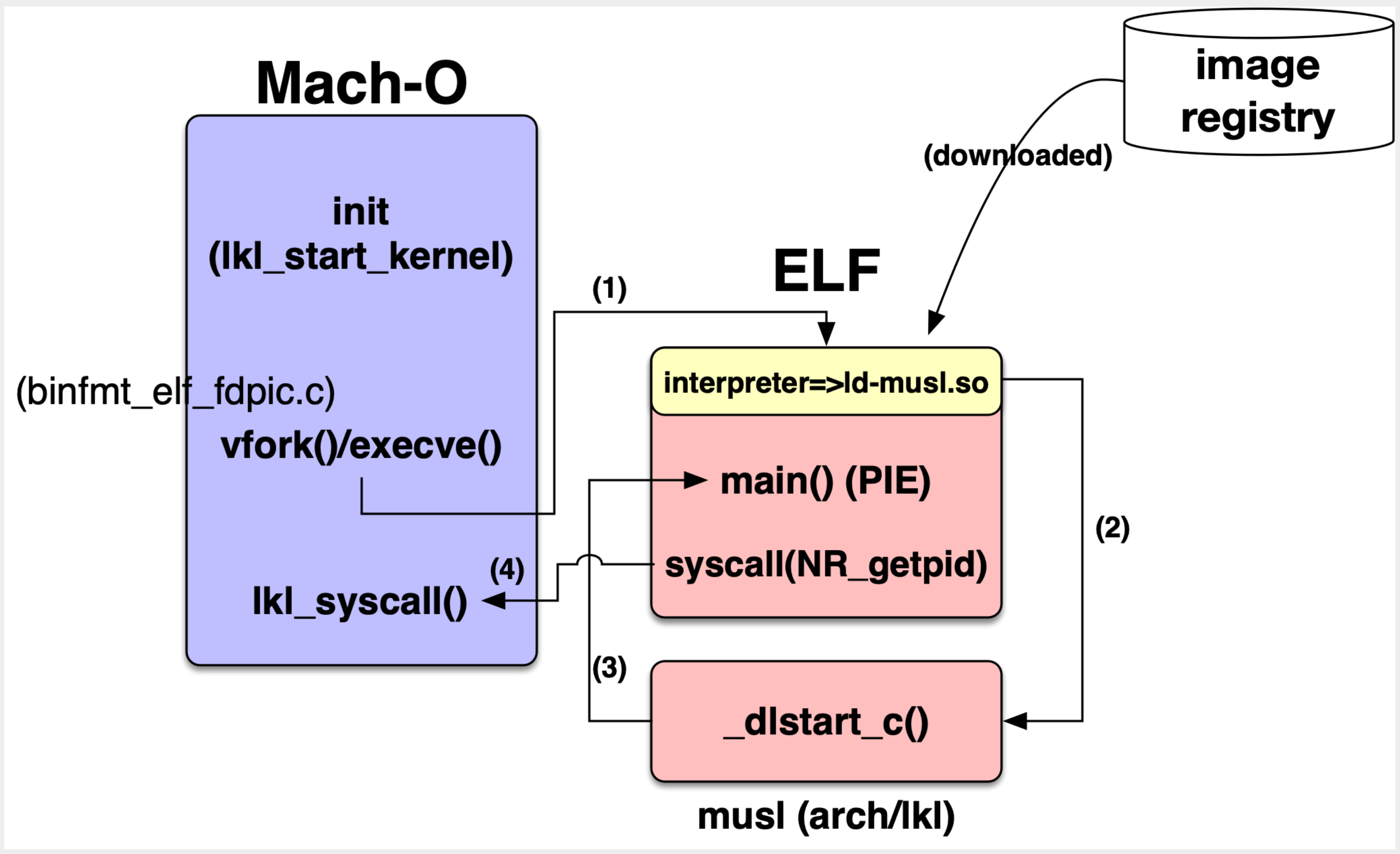
DIGEST	OS/ARCH	COMPRESSED SIZE ⓘ
9bb13ae8c65f	darwin/amd64	2.01 MB
5554575c9ce5	linux/amd64	1.82 MB
94270caa6b97	linux/arm/v7	1.67 MB
0ebdb37bb685	linux/arm64	1.79 MB

Demo: alpine linux on macOS



https://asciinema.org/a/347292

Docker for mac+ : How LKL works



0. (Mach-O) Run LKL as init process
1. (Mach-O) (v)fork/execve Linux ELF binary
2. (ELF) interpreter (musl+) loads (downloaded) ELF program
3. (ELF) call main() function
4. (ELF) syscall => LKL syscall (libc replacement)
5. (Mach-O) handle lkl syscall from ELF

Limitations

- vfork (nommu)
  - still bugs
  - has to block parent process until children exit
- no glibc-based image support (will work on)
- libc-replacement doesn't work with static binaries

Summary

- containerd port for darwin (PR under review)
  - https://github.com/containerd/containerd/pull/4526
- Run Linux applications on macOS without Hypervisor.framework
  - not exactly, but WSL1-like
- dockerd port will follow after containerd upstream

References

- pull request
  - https://github.com/containerd/containerd/pull/4526
- Linux kernel library (LKL)
  - https://github.com/lkl/linux
- runu (OCI runtime for LKL)
  - https://github.com/ukontainer/runu