Running Docker (and more) in NetBSD via Lima

Leonardo Taccari
<leot@NetBSD.org>

The NetBSD Foundation

EuroBSDCon 2025 - NetBSD Summit
September 26th 2025
Zagreb, Croatia

Outline

Lima

Lima: **Li**nux **Ma**chines

- Lima stands for **Li**nux **Ma**chines
- Original goal was to promote `containerd` to Mac users
- Deploy virtual machines with automatic file sharing and port forwarding
- Supports several Linux distributions as *guests*
- Supports several *hosts*: macOS, Linux, NetBSD and DragonFly BSD
- Daemonless tool: `limactl`

Lima: How it works? [1]

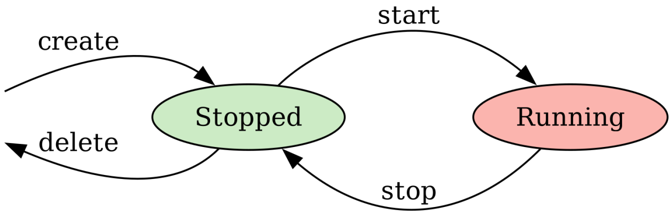
Virtualization	QEMU
Networking	user-mode emulation (AKA slirp)
Filesystem mounts	QEMU virtfs and 9P
Host and guest agents	Permit several communications between host and guest (e.g. automatic port forwarding)

[1] We will focus only on Lima on NetBSD and main features/defaults from now on. Please consult the official Lima documentation for more.

Lima: Defining instances: templates

- Guests are defined via templates
- Templates are written in YAML
- Permit to set CPUs, memory, disk, provision scripts and more
- Lima provides dozens of templates for several distros and versions of them (e.g. Alpine, Debian, Ubuntu, Fedora, etc.)
- Templates can be customized
- Every instance is created based on a template

Lima: Instance lifecycle



Lifecycle of Lima instances and status transactions via `limactl` commands.

Demo

Demo: Writing a Docker template based on Alpine I

```
minimumLimaVersion: 1.1.0

base:
  - template://alpine

containerd:
  system: false
  user: false

provision:
- mode: system
  script: |
    #!/bin/sh
    sed -i 's/host.lima.internal.*/host.lima.internal host.docker.internal/' /
         etc/hosts
- mode: system
  script: |
    #!/bin/sh
    set -eux -o pipefail
    command -v docker >/dev/null 2>&1 && exit 0
    apk update
    apk add docker
    rc-update add docker default
    service docker start
- mode: user
  script: |
    #!/bin/sh
    set -eux -o pipefail
    sudo addgroup "$USER" docker
probes:
```

Demo: Writing a Docker template based on Alpine II

```yaml
- script: |
    #!/bin/sh
    set -eux -o pipefail
    if ! timeout 30s bash -c "until command -v docker >/dev/null 2>&1; do sleep 3; done"; then
      echo >&2 "docker is not installed yet"
      exit 1
    fi
hostResolver:
  hosts:
    host.docker.internal: host.lima.internal
portForwards:
- guestSocket: "/var/run/docker.sock"
  hostSocket: "{{.Dir}}/sock/docker.sock"
message: |
  To run `docker` on the host (assumes docker-cli is installed), run the following commands:
  ------
  docker context create lima --docker "host=unix://{{.Dir}}/sock/docker.sock"
  docker context use lima
  docker run hello-world
  ------
```

Demo: Creating instance: `limactl create` I

- To create the instance we run:

  ```
  $ limactl create \
      --name=docker \
      --cpus=2 \
      --memory=4 \
      --disk=20 \
      .../alpine-docker.yaml
  ```
- This will fetch the image and populate the instance directory `~/.lima/docker`

Demo: Creating instance: `limactl create` II

Demo!

Demo: Running the instance: `limactl start` I

- To start the instance we run:

```
$ limactl start docker
```

- In the first boot it will also initialize it (run cloud-init, grow partition, reboot) and it will take more time

Demo: Running the instance: `limactl start` II

Demo!

Demo: Using Docker (and more) I

- To verify that Docker is working we run the hello-world container:

```
$ docker run hello-world
```

Demo: Using Docker (and more) II

Demo!

Demo: Using Docker (and more) III

- Let's run something a bit more complex, nginx by sharing the mount and exposing its port to host's port 8080:

```
$ docker run -v \
    ./static-html-directory:/usr/share/nginx/html:ro \
    -p 8080:80 \
    nginx
```

Demo: Using Docker (and more) IV

Demo!

Demo: Using Docker (and more) V

- Something even more complex... Kubernetes via kind!:

```
$ kind create cluster
```

Demo: Using Docker (and more) VI

Demo!

Demo: Stopping the instance: `limactl stop` I

- To stop the instance we run:

  ```
  $ limactl stop docker
  ```
- This will shutdown the instance.
- We can start it again via `limactl start`.

Demo: Stopping the instance: `limactl stop` II

Demo!

Demo: Deleting the instance: `limactl delete` I

- To delete the instance we run:

```
$ limactl delete docker
```

- This will delete the instance so also all its data is lost, beware!

Demo: Deleting the instance: `limactl delete` II

Demo!

Status, possible TODOs, help needed

- Everything packaged in pkgsrc-wip as `wip/lima`, `wip/docker-cli` and `wip/kind`, probably ready to be imported after pkgsrc-2025Q3
- Ubuntu, Ubuntu LTS and Debian images fails with `Kernel panic - not syncing:  IO-APIC + timer doesn't work!  Boot with apic=debug and send a report. Then try booting with the 'noapic' option.` [2]
- Testing on other platforms appreciated! (in particular macOS)

[2] I think we have dealed with something similar for `sysutils/podman`!

Thanks

- Particular thank you: `<maya>`, `<riastradh>`, `<ryoon>`, `<cirnatdan>` (and I'm sure I have missed a lot of other folks, sorry!)

Conclusions

- We have introduced Lima: **Li**nux **Ma**chines
- We have learned how to write/extend a custom Lima template to provision and configure packages in the guest
- We have seen how to run Docker on NetBSD via Lima with volume sharing and automatic port forwarding

References

Lima Authors.
Lima homepage.
https://lima-vm.io/, a.

Lima Authors.
Lima documentation.
https://lima-vm.io/docs/, b.

Docker Inc.
Docker Docs.
https://docs.docker.com/.

The Kubernetes Authors.
kind homepage.
https://kind.sigs.k8s.io/.

Questions?