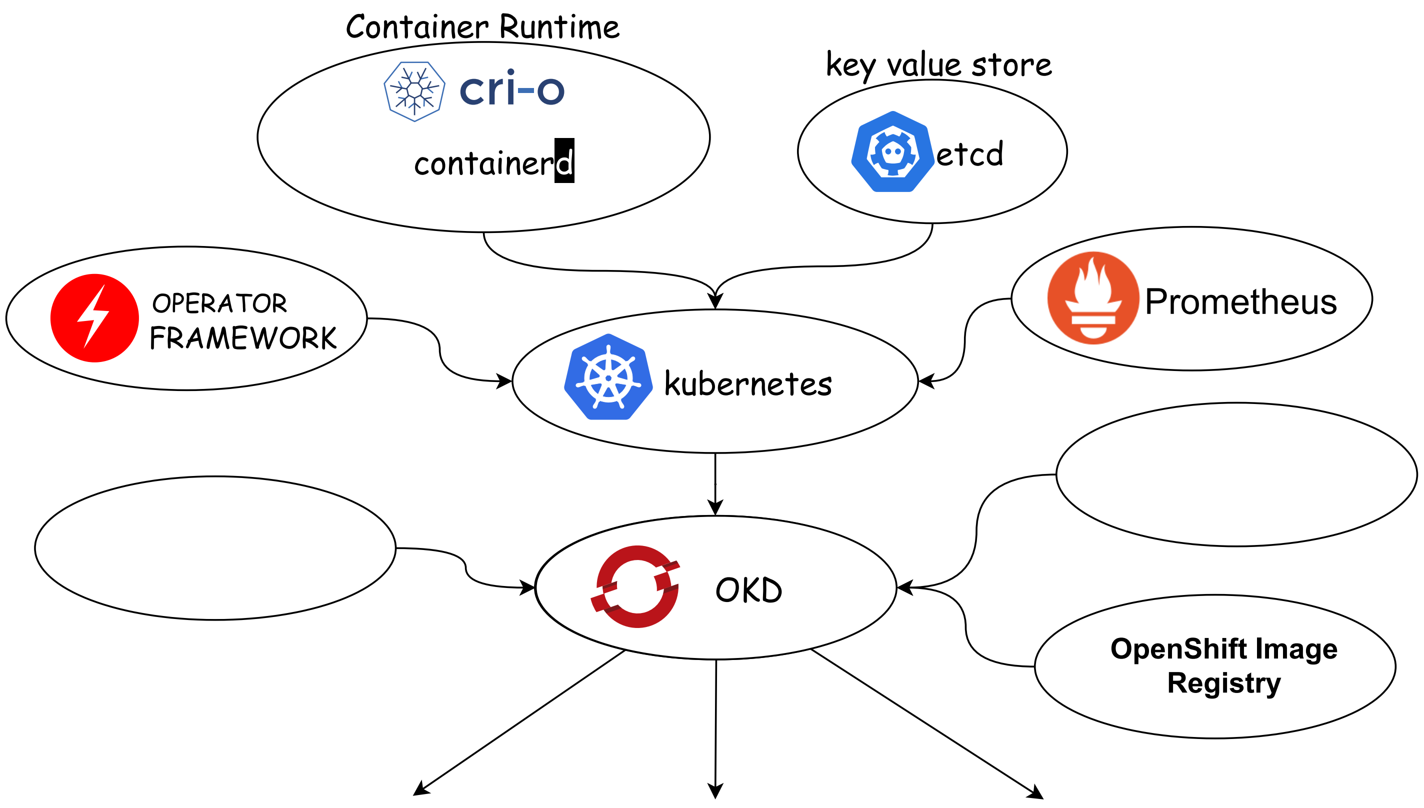


Red Hat OpenShift Container Platform

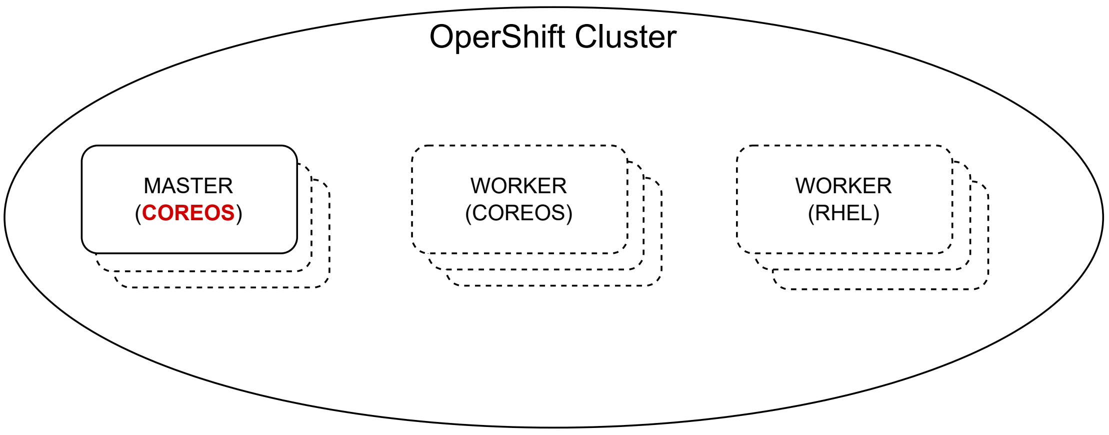
- Public/private DC.
- Bare metal and multiple cloud and virtualization providers.
- Full control by customer.

Red Hat OpenShift Dedicated

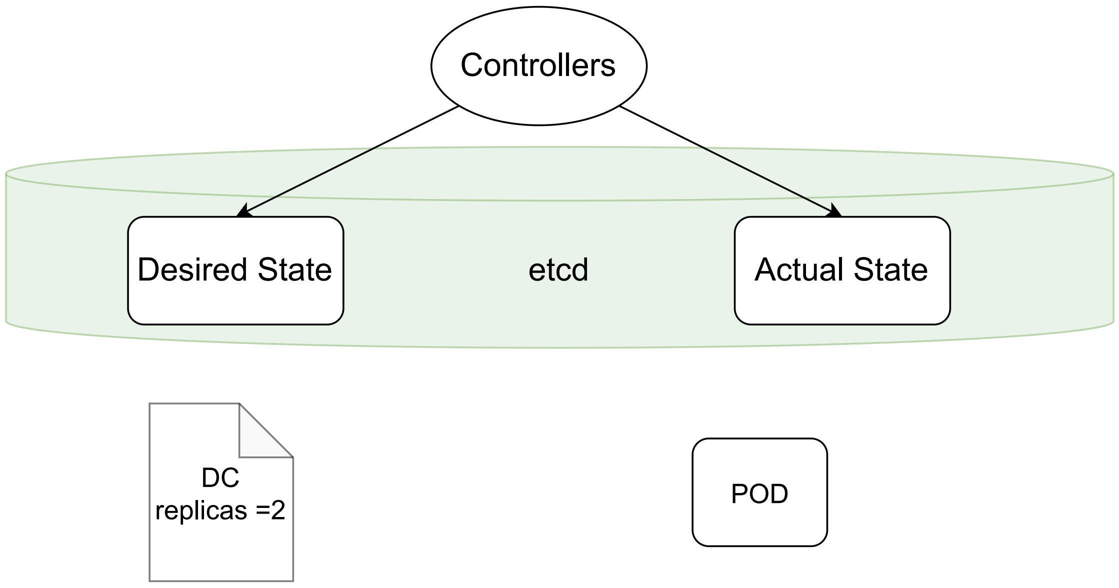
- Managed cluster in public cloud.
- RH manages the cluster.
- Customer manages updates and add-on services.

Red Hat OpenShift Online

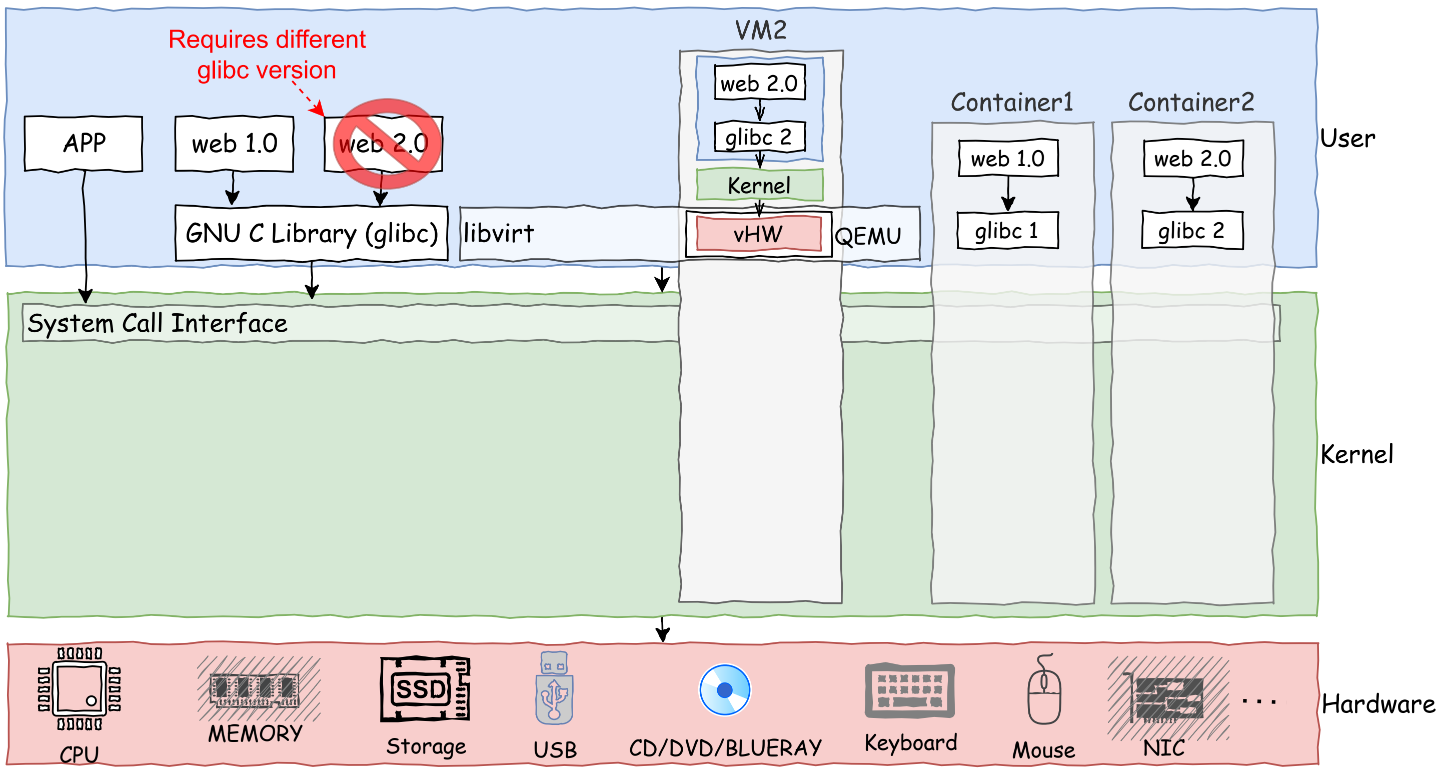
- Public hosted cluster.
- Shared resources by multiple customers.
- RH manages cluster life cycle.



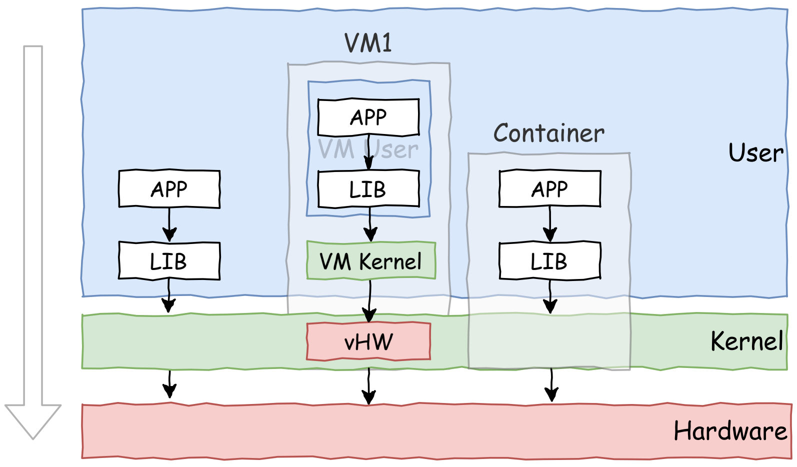
Kubernetes Declarative Architecture



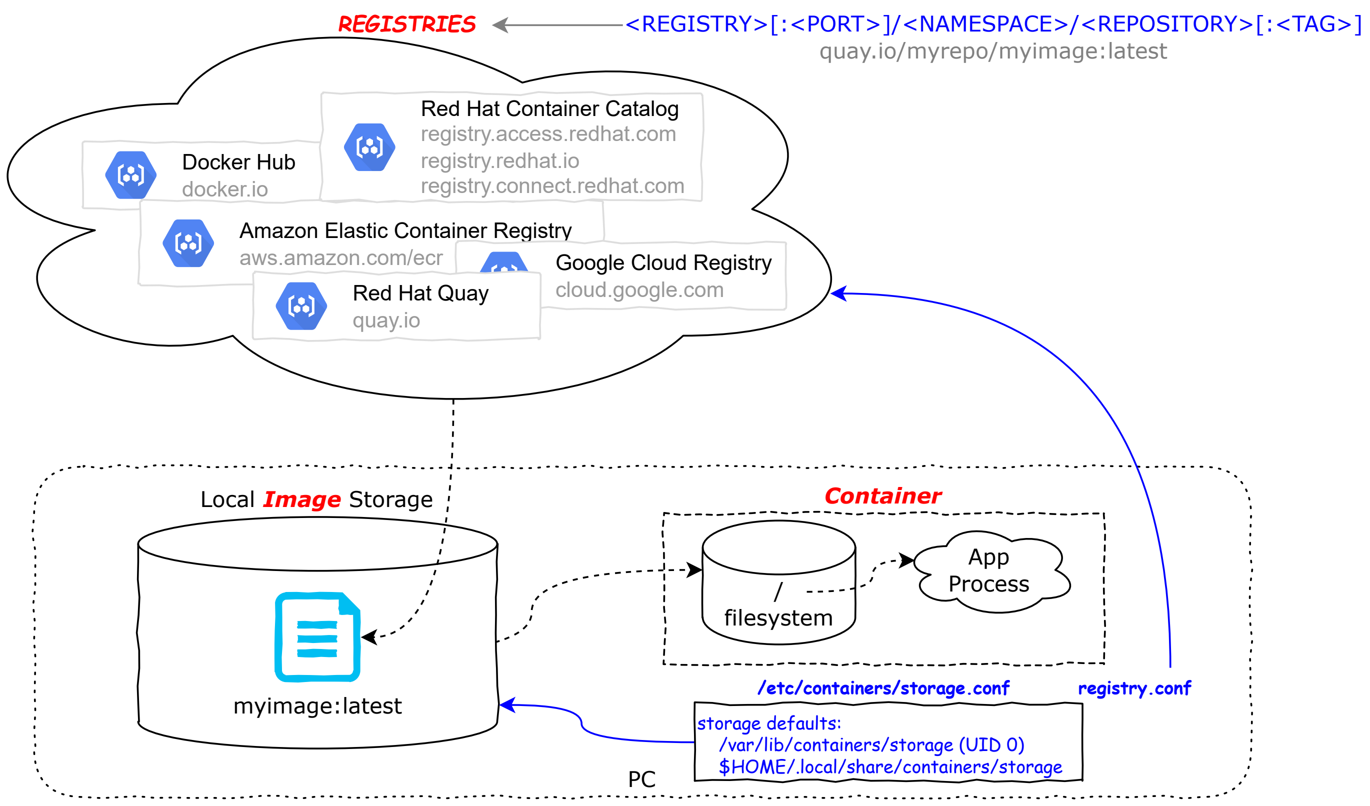
VM vs Container



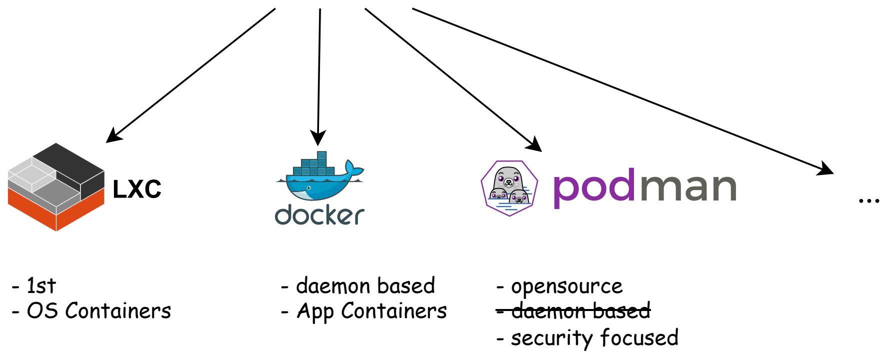
Ref: https://www.redhat.com/en/blog/all-you-need-know-about-kvm-userspace
https://www.packetcoders.io/what-is-the-difference-between-qemu-and-kvm/



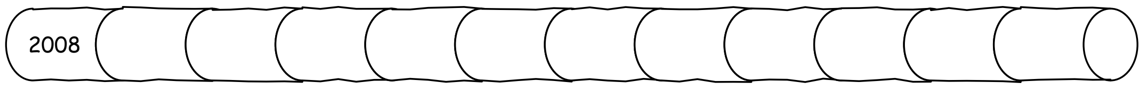
Container Architecture


REGISTRIES
<REGISTRY>[:<PORT>]/<NAMESPACE>/<REPOSITORY>[:<TAG>]
quay.io/myrepo/myimage:latest
Red Hat Container Catalog
registry.access.redhat.com
registry.redhat.io
registry.connect.redhat.com
Docker Hub
docker.io
Amazon Elastic Container Registry
aws.amazon.com/ecr
Google Cloud Registry
cloud.google.com
Red Hat Quay
quay.io
Local Image Storage
myimage:latest
Container
/
filesystem
App Process
/etc/containers/storage.conf
registry.conf
storage defaults:
/var/lib/containers/storage (UID 0)
$HOME/.local/share/containers/storage
PC

Container Utilities



OS Container Vs Application Containers



https://developer.ibm.com/tutorials/multi-architecture-cri-o-container-images-for-red-hat-openshift/

http://www.haifux.org/lectures/299/netLec7.pdf
https://kernelnewbies.org/Linux_2_X_XX

```
hostname
unshare -u
hostname abc
hostname
exit
```

Heading

man 7 namespaces

- mount (2.4.19) - 3/8/2002 - CAP_SYS_ADMIN
- pid (2.6.24) - 24/1/2008
- net (2.6.29) - 23/3/2009
- ipc (2.6.19) 29/11/2006
- uts (2.6.19) 29/11/2006
- user (3.8) 18/2/2013 no cap
- cgroup (4.6) 15/5/2016
- time - 3/2020

Podman

Image and Registry Operations

`podman login [-u USER] [-p PASS] [REGISTRY]`	Only if required. Accessing private repo or updating image.
`podman logout {-a | REGISTRY}`	Logout of registry (-a for all).
`podman images [-q]`	List local images (-q only show id).
`podman rmi IMAGE...`	Removes local image(s). Use -af with caution.
`podman search KEYWORD`	Search registry for an image.
`podman pull SOURCE`	Pull image from a registry.

Where,

SOURCE	`[REGISTRY[:PORT]/NAMESPACE/]IMAGE[:TAG]`
	`dir:PATH`
	`docker-archive:PATH`
	`oci-archive:PATH`

`podman tag IMAGE[:TAG] TARGET_NAME[:TAG]`	Add an additional name to a local image
`podman push IMAGE`	Upload an image to the registry

Container Operations

```
podman run [--name NAME] [-p PORT_INFO] [-v VOL_INFO] [-d] [-it] IMAGE [CMD_INFO]
```

Where,

--name *NAME*	Container name. Autogenerated if not provided.
-p *PORT_INFO*	**[** *LOCAL_IP* **:]** *LOCAL_PORT* **[[:** *CONT_IP* **] :** *CONT_PORT* **]** Mapping between local IP:PORT to container IP:PORT
-v *VOL_INFO*	*LOCAL_DIR* **:** *CONT_DIR* Mapping between local dir to container dir.
-d	Run in detached mode (background).
-it	-i keep stdin open, -t allocate a pseudo-tty.
IMAGE	Image used to create the container.
CMD_INFO	*CMD* **[***ARG...***]** Command to run in container.

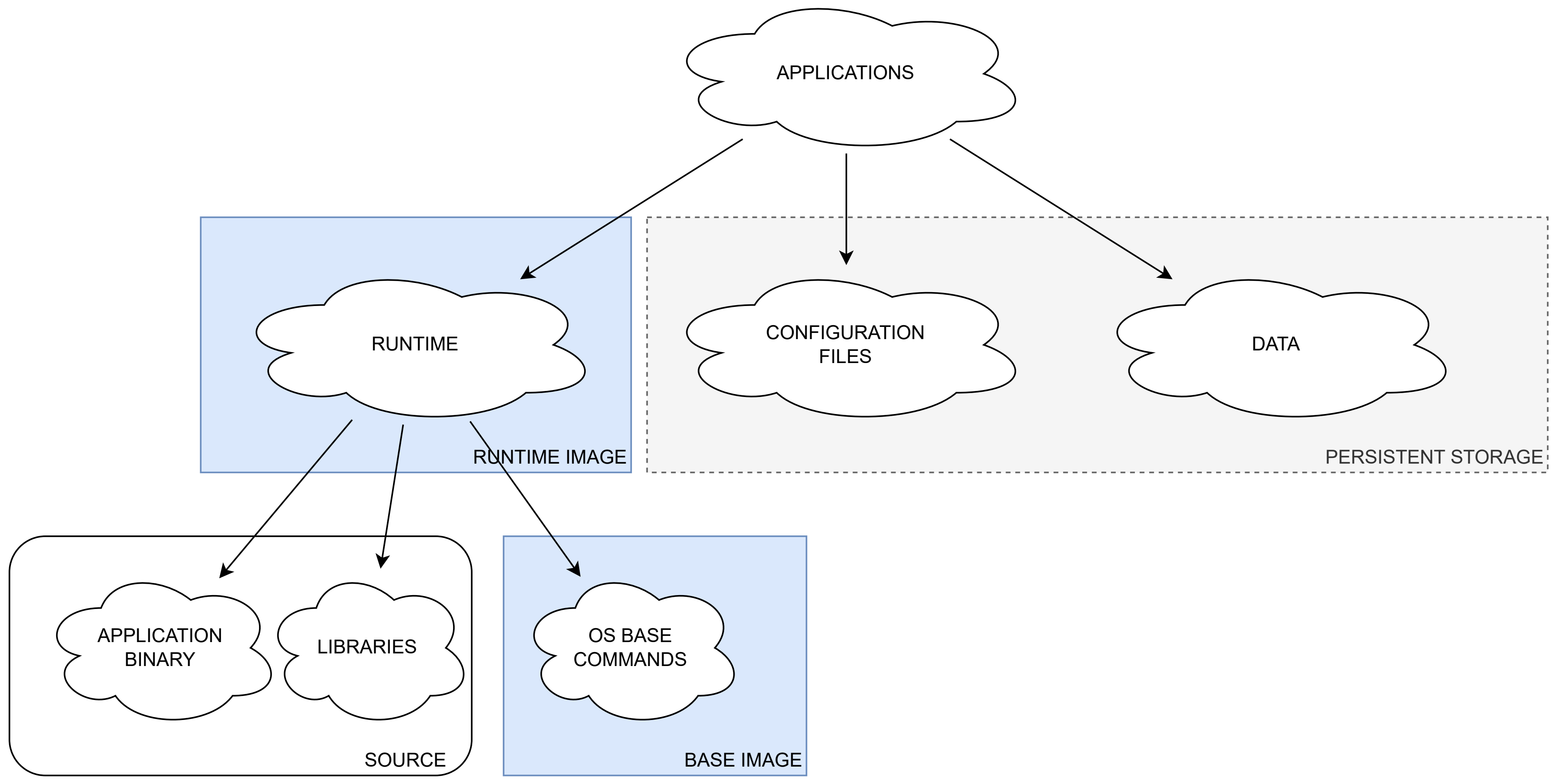
`podman exec [-it] CONTAINER CMD_INFO`	Execute command inside running container.
`podman ps [-a] [-q]`	List containers (-a for all, -q only show container id).
`podman rm CONTAINER...`	Remove one or more stopped containers. (-f includes running and paused containers).
`podman start|stop|restart CONTAINER...`	Start, stop or restart one or more containers.
`podman kill [-s SIGNAL] CONTAINER...`	Send signal to one or more containers.

For more info:

`podman --help` OR `man podman`

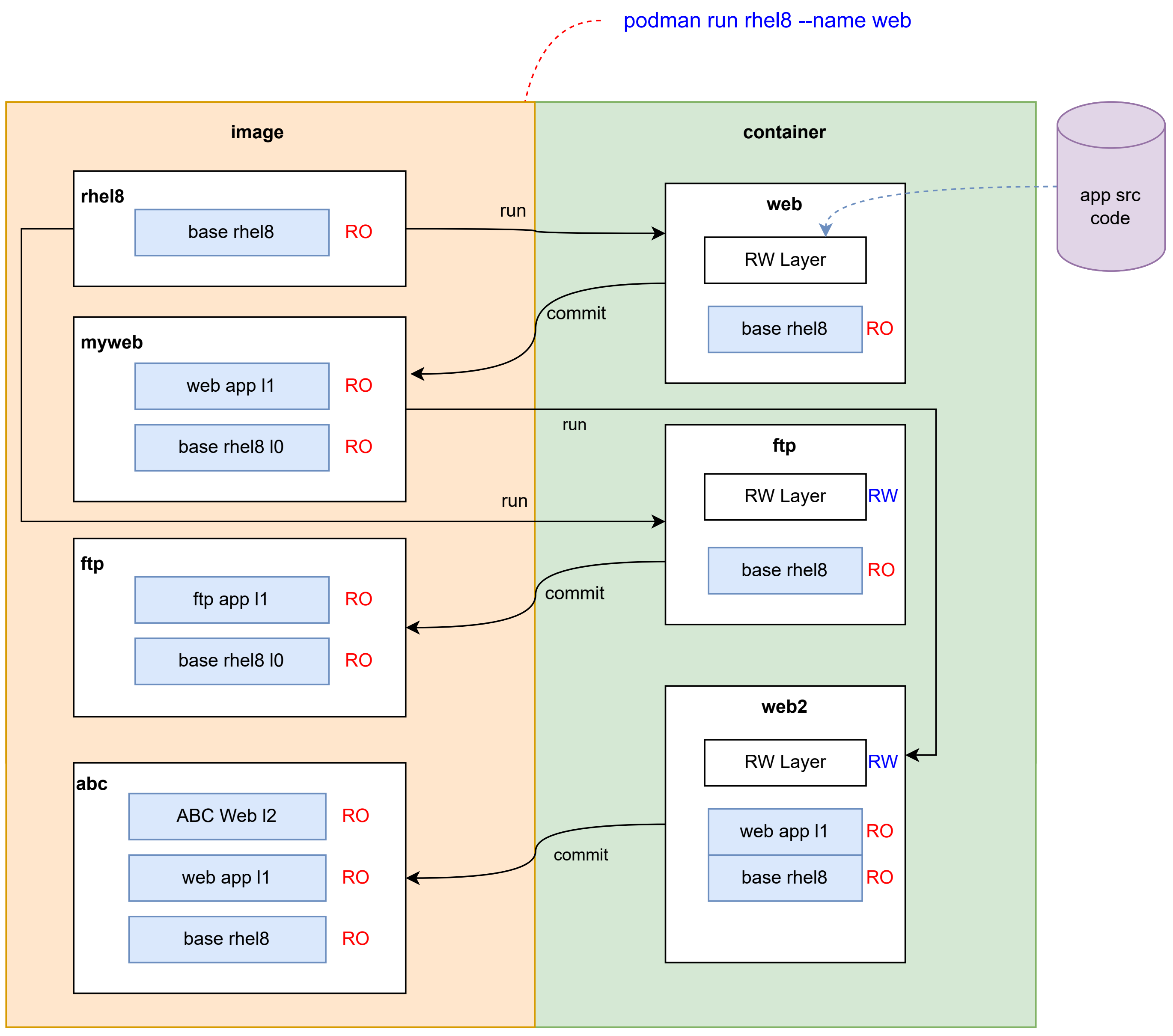
Each sub command has it's own man page. i.e man podman-run, man podman-images, etc.

Basic Container Design

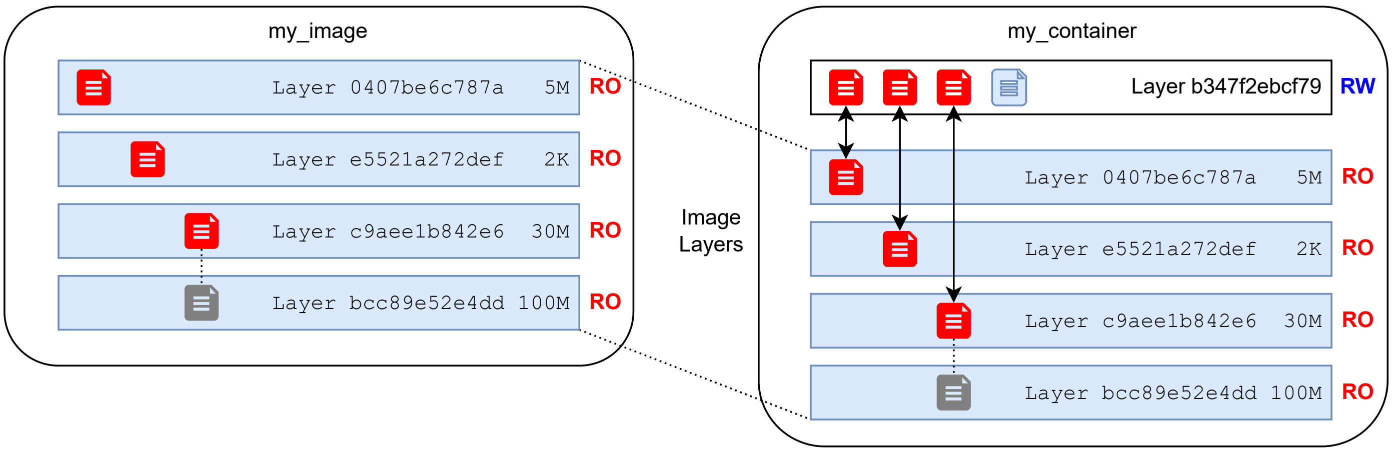


Creating Image

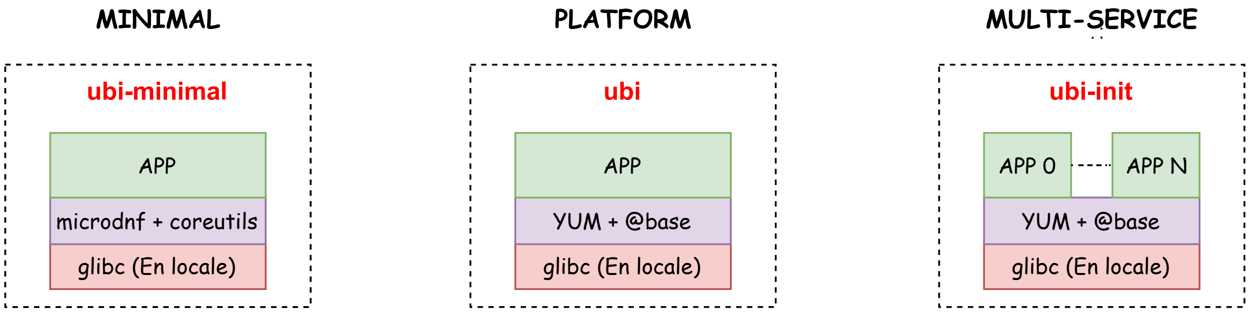
1. Manual
2. Dockerfile/Containerfile
3. Source-To-Image(s2i/STI)
    a) get runtime image and create container
    b) clone source code into container
    c) compile source code
    d) deploy/publish compiled app
    e) cleanup
    f) save container as image



UnionFS - A Stackable Unification File System



BASE IMAGE TYPES



Designed for apps that contain their own dependencies (Python, Node.js, .NET, etc.)

- Minimized pre-installed content set
- no suid binaries
- minimal pkg mgr (install, update & remove)

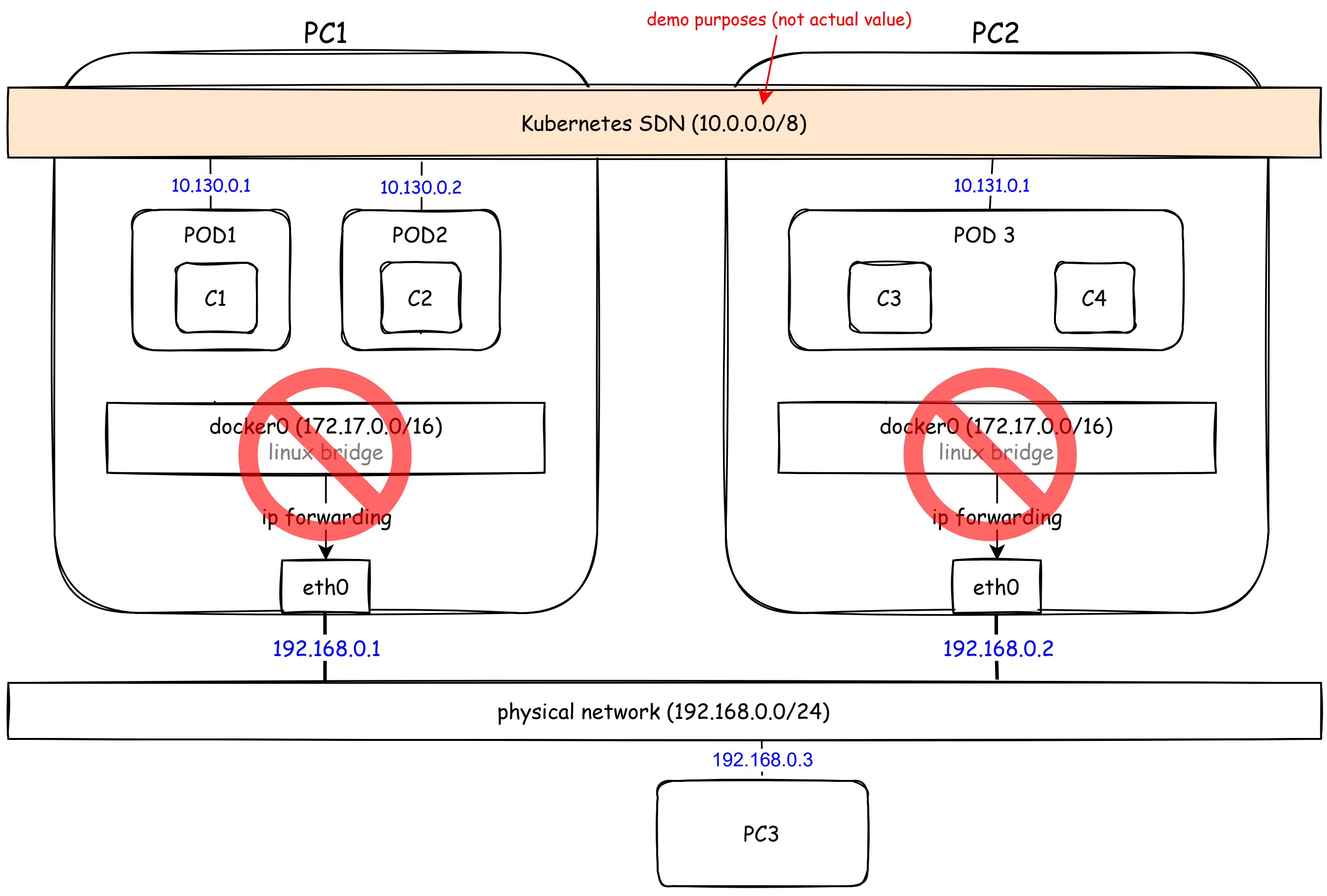
For any apps that runs on RHEL

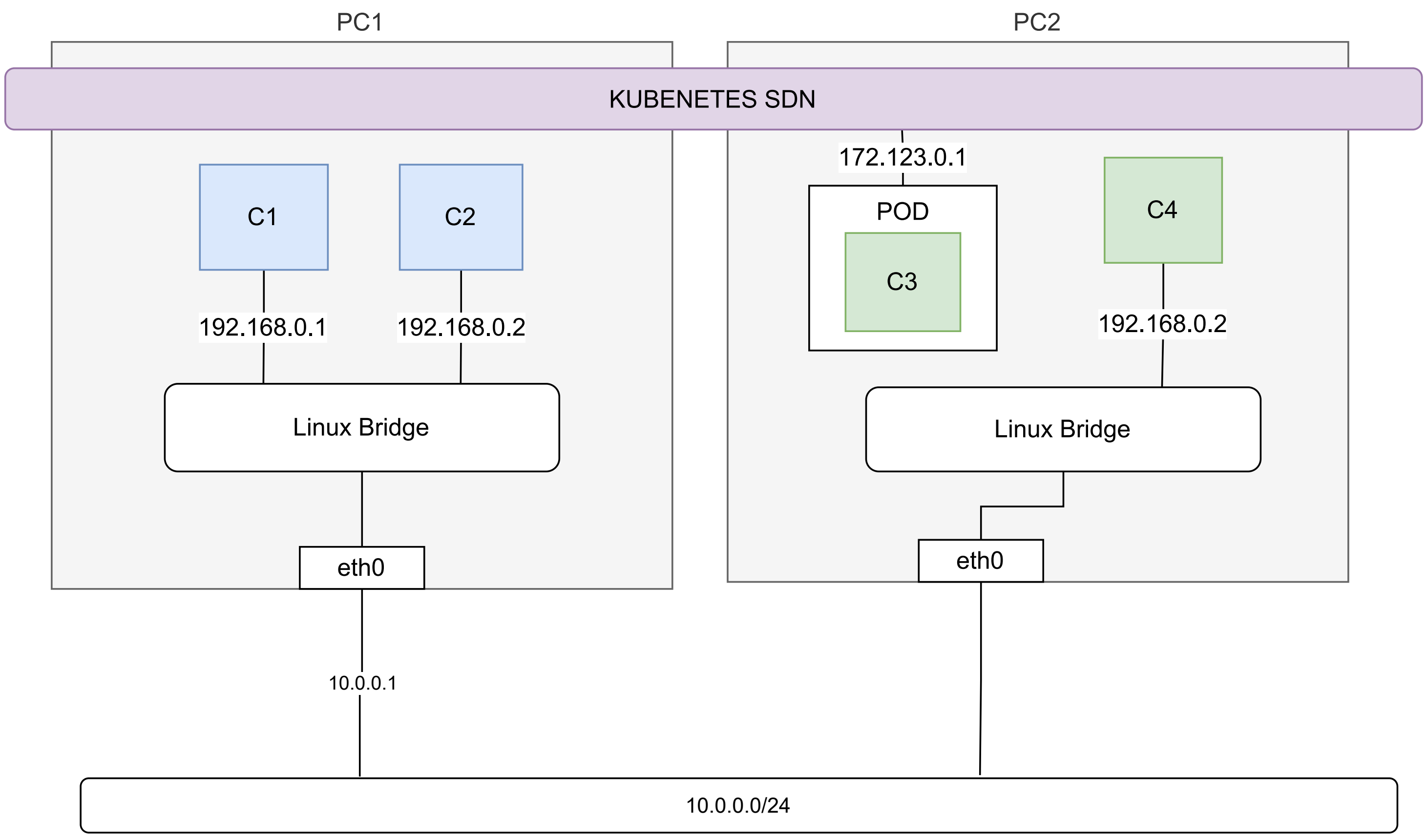
- Unified, OpenSSL crypto stack
- Full YUM stack
- Includes useful basic OS tools (tar, gzip, vi, etc)

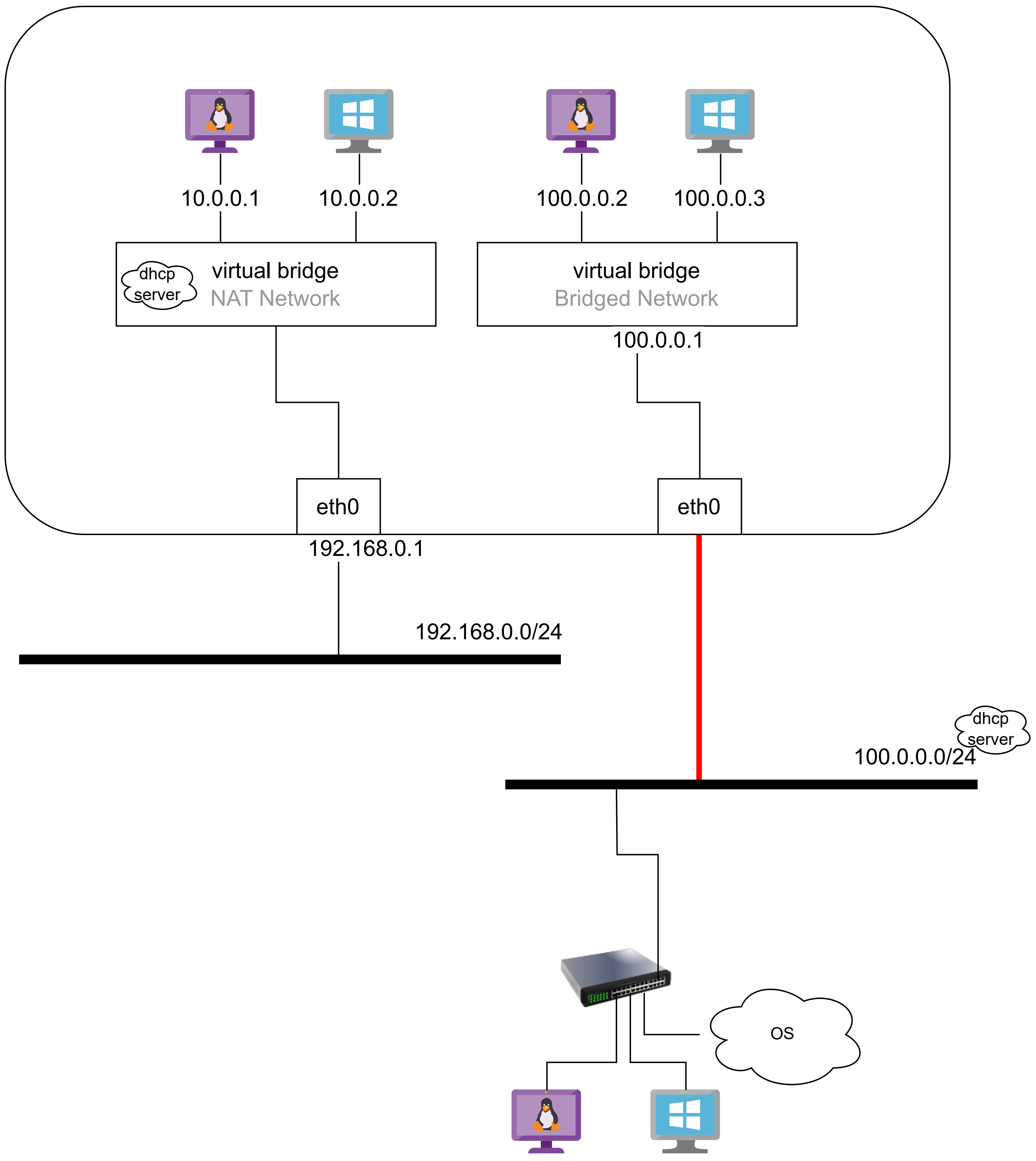
Eases running multi-service in single container

- configured to run systemd on start
- allows you to enable th services at build time

Basic Network - Container vs Kubernetes






10.0.0.1
10.0.0.2
100.0.0.2
100.0.0.3
dhcp server
virtual bridge
NAT Network
virtual bridge
Bridged Network
100.0.0.1
eth0
eth0
192.168.0.1
192.168.0.0/24
dhcp server
100.0.0.0/24
os