DOCKRIZE!

BELAID MOA

COMPUTE CANADA/WESTGRID/UNIVERSITY OF VICTORIA

bmoa@uivc.ca

AGENDA

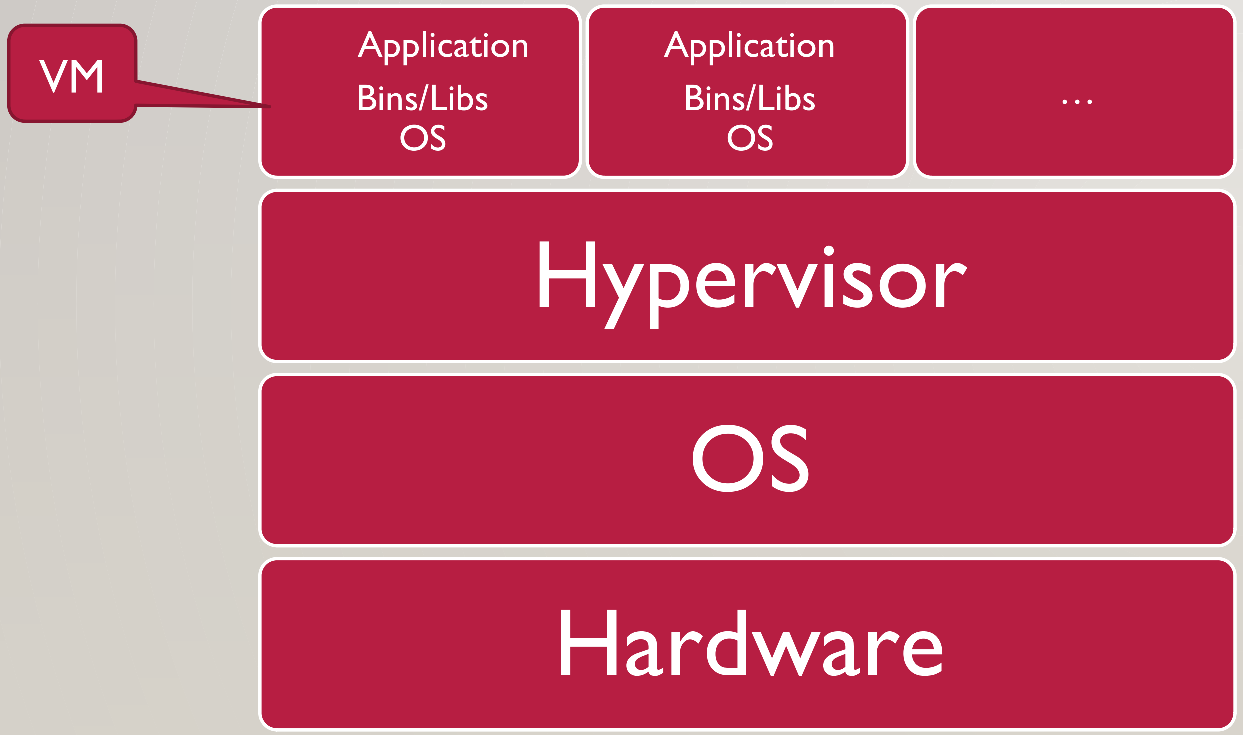
- Why dockers/containers?
- Do researchers/scientist really need dockers?
- Why not VMs?
- The spectrum of deployments
- Docker Elements
- Docker Workflow
- From Docker to HPC

WHY DOCKERS/CONTAINERS?

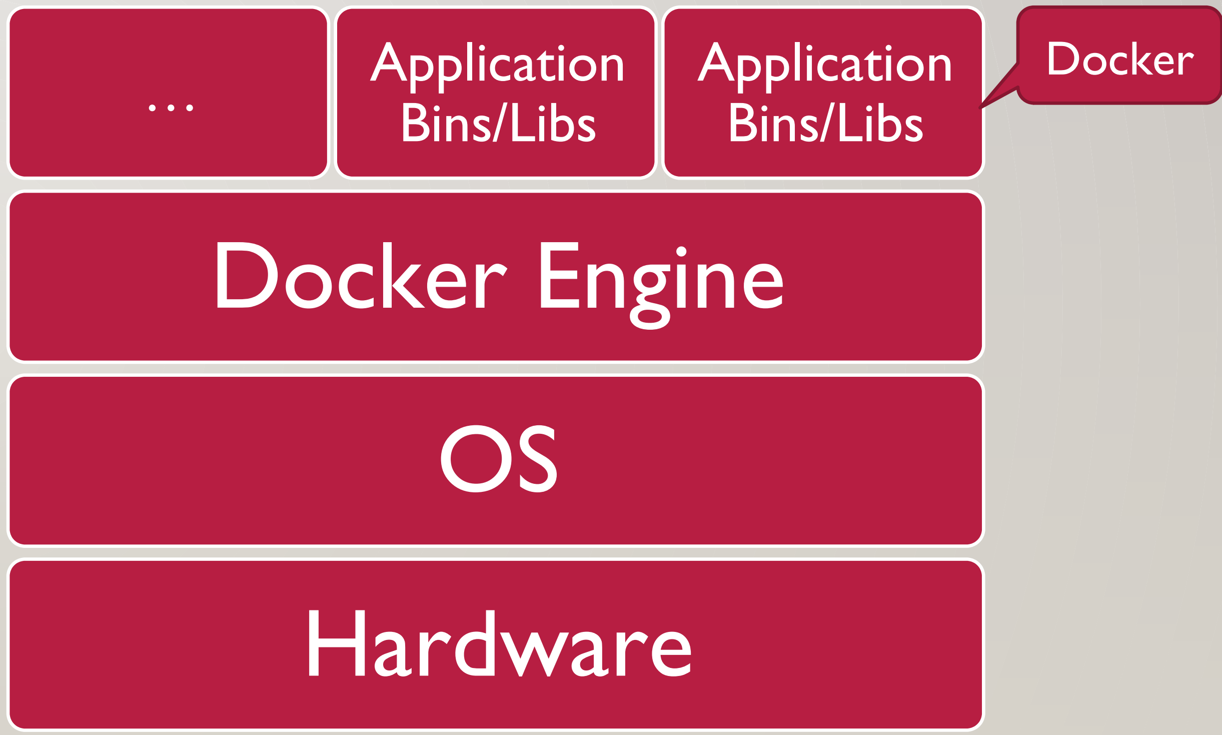
- Building, deploying, sharing, and running applications should be easy
  - "What? It is working on my machine!"
- Updating the hosts should not break the applications running on them
  - The matrix of hell!
- Getting the application up and running should be quick
- Packaging and shipping the application should not too hard
- Isolating different applications and their dependencies from each other
- Dev/Test/Prod environments should not matter

But Isn't virtualization enough? ….

WHY NOT VMS?



Virtualization

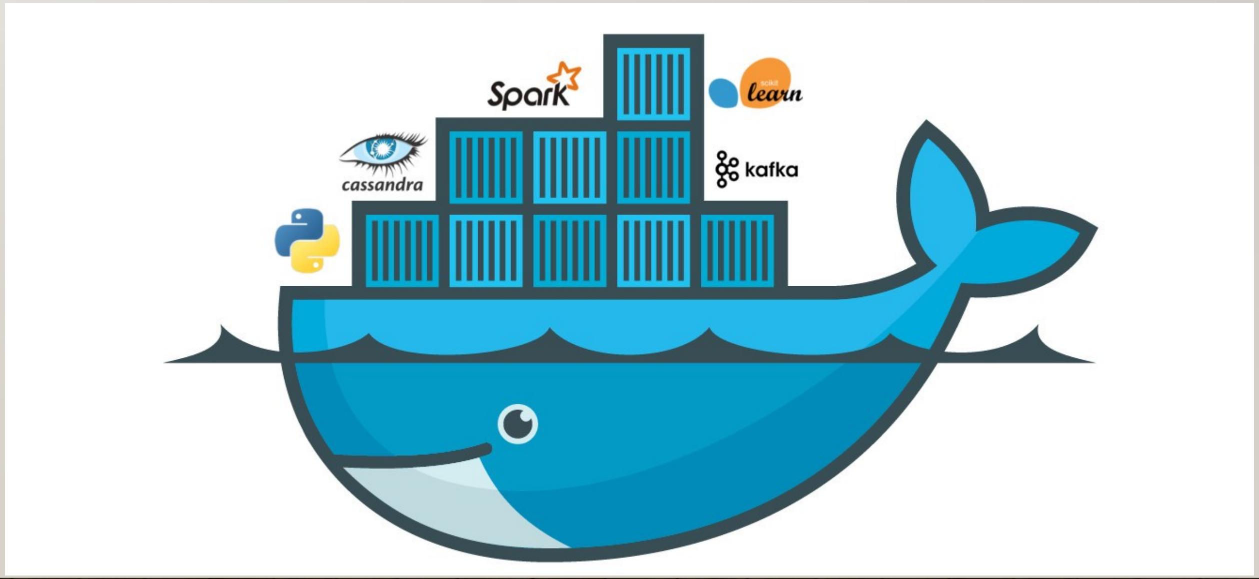


Dockerization

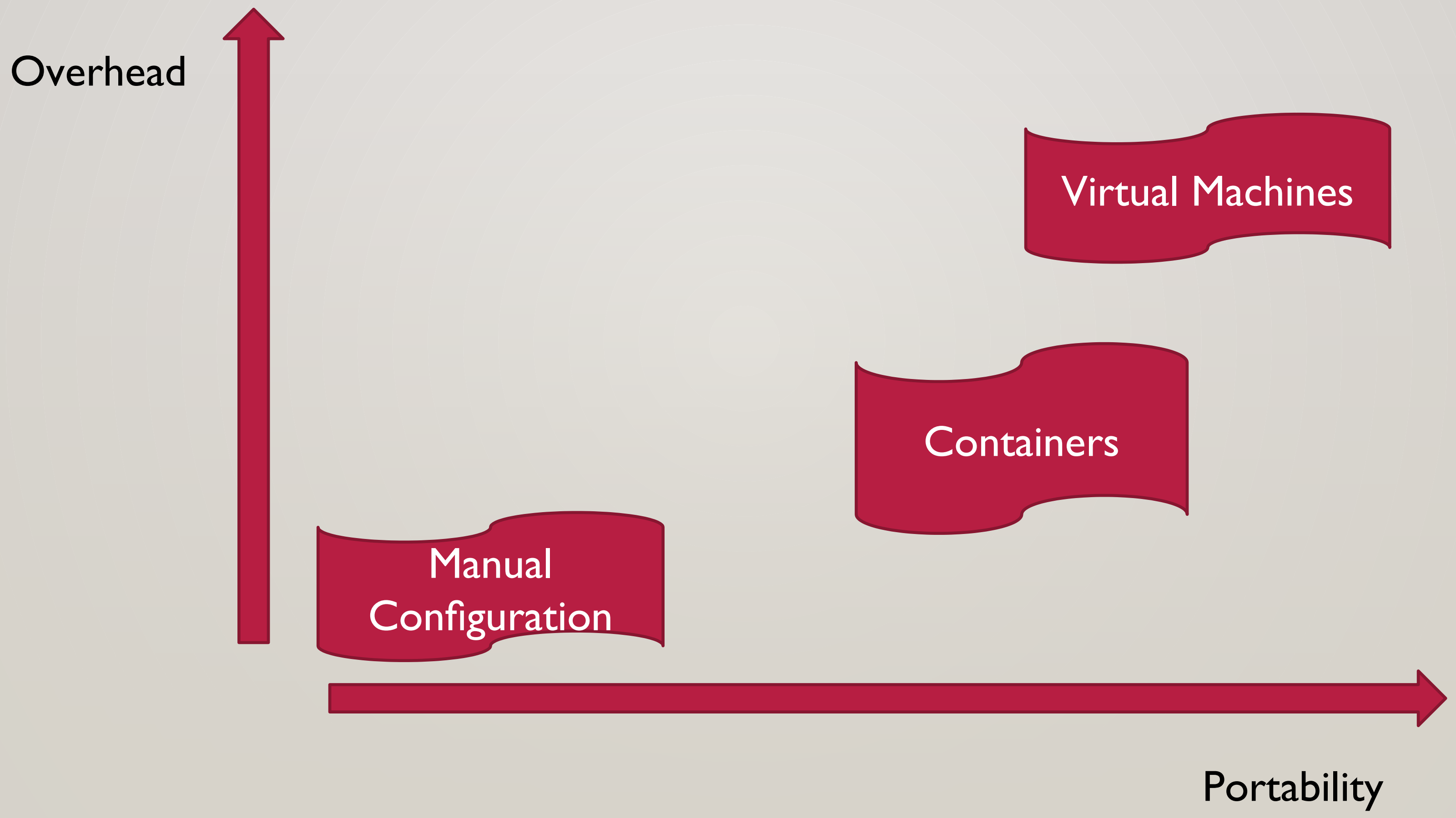
DO WE REALLY NEED CONTAINERS/DOCKERS?

- Researchers are interested in
  - long term reproducibility of results
    - their 10 year old applications must work
  - getting their jobs to run ASAP
  - using computational resources wherever they are available
- Containers are here to the rescue!
  - Package your application into a container and get it deployed anywhere, anytime
- Real stories
  - Magpy for geomagnetism
  - Mountain Legacy Project
  - Cyberhubs
  - Substance
  - BESOS

docker


Spark
scikit learn
cassandra
kafka

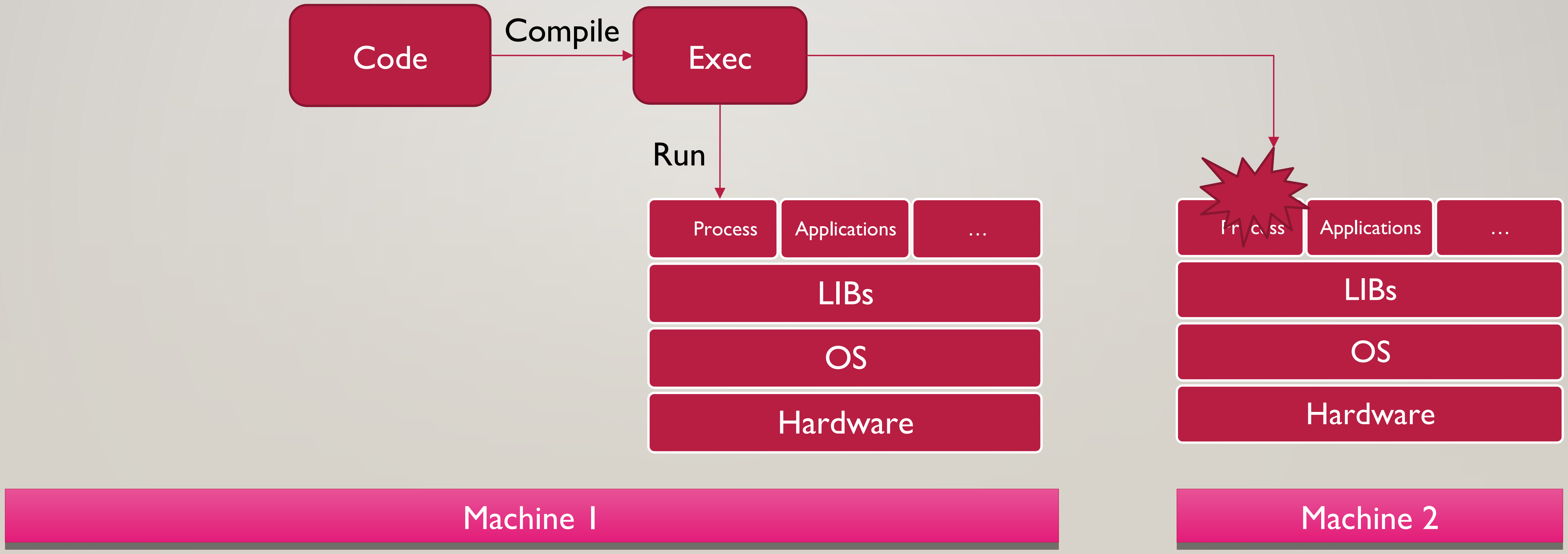
SPECTRUM OF APPLICATION DELPOYMENT



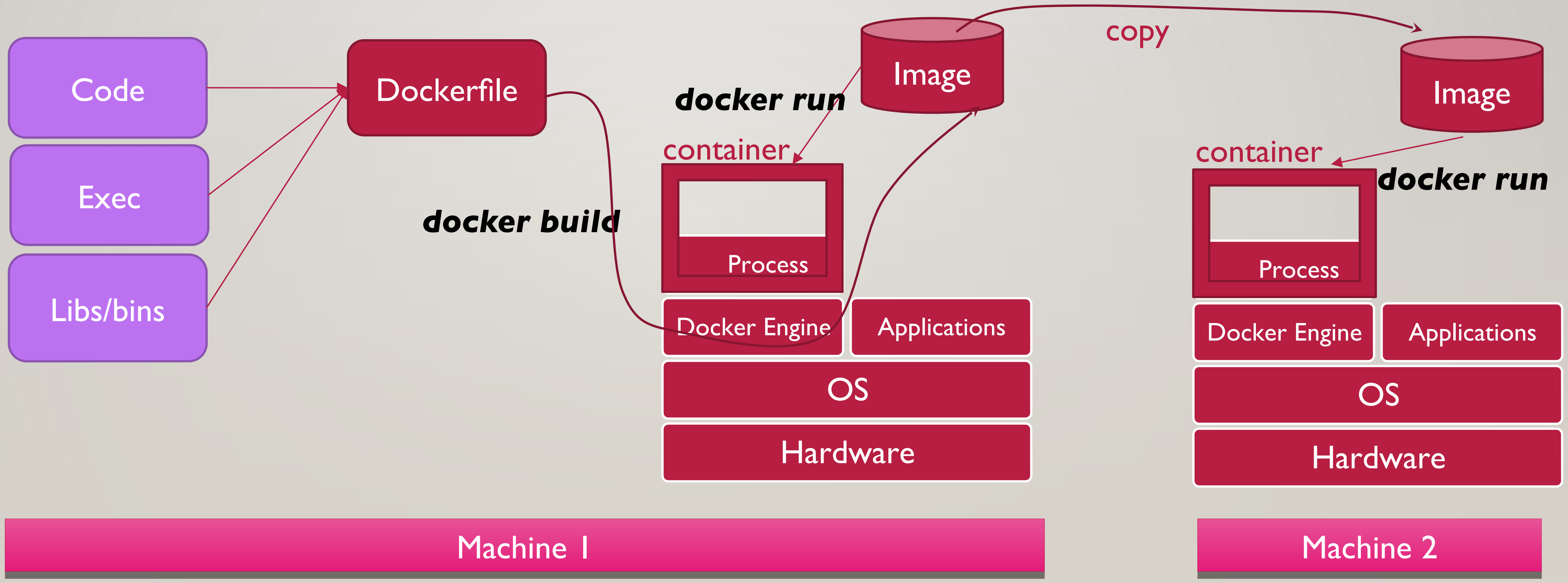
WHY DOCKERS/CONTAINERS

- Flexible: Unfriendly and complex applications can be easily dockerized.
- Lightweight: Containers share and leverage the host kernel, and therefore they are much more efficient than virtual machines.
- Portable: build locally and deploy anywhere.
- Loosely coupled: Containers are highly self sufficient and encapsulated, allowing you to replace or upgrade one without disrupting others.
- Scalable: You can increase and automatically distribute container replicas across a datacenter.
- Secure: Containers apply aggressive constraints and isolations to processes without any configuration required on the part of the user.

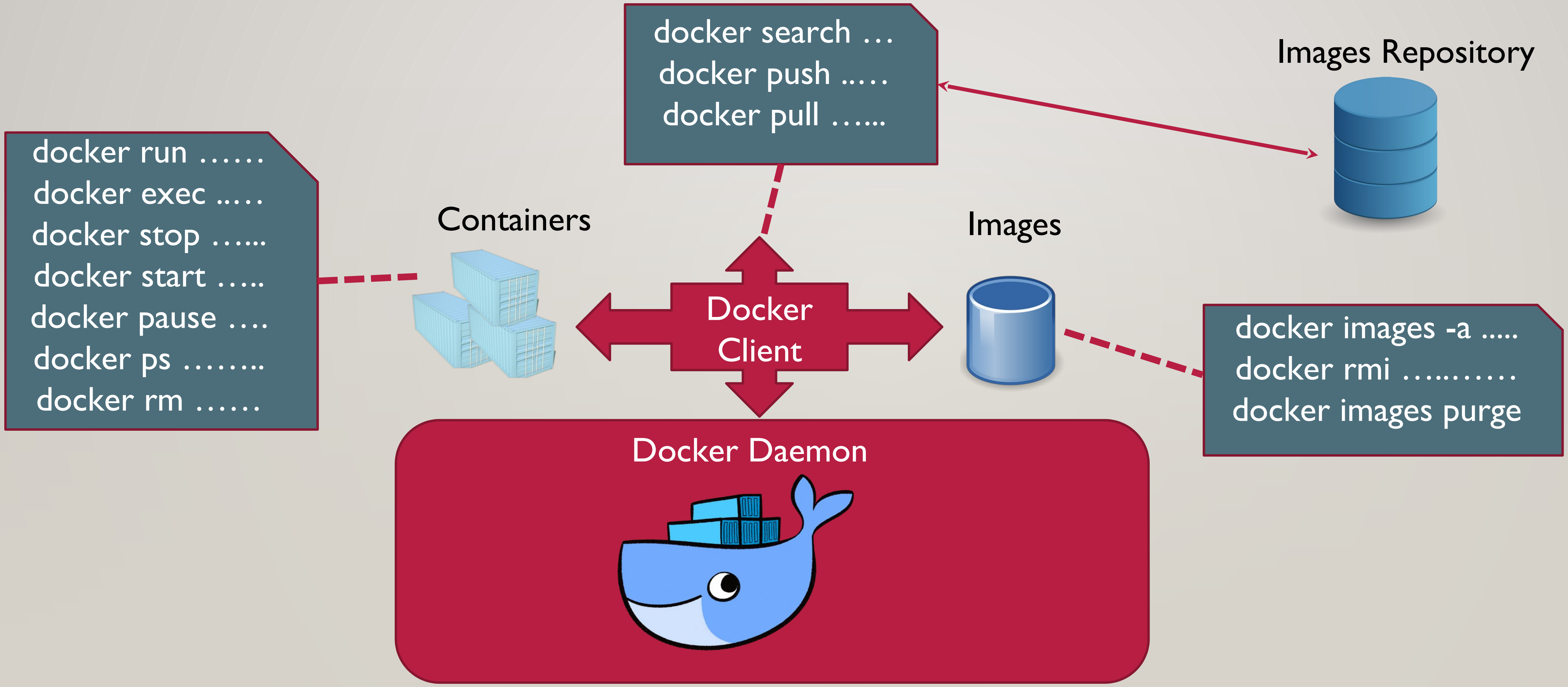
DOCKER WORKFLOW



DOCKER WORKFLOW



DOCKER COMPONENTS



DOCKER COMPONENTS

- Images

  They play similar role as VM images. There are templates, self-contained packages and read-only layers for running containers – they have everything you need to run your application. Usually each image fulfills a single task, and is generated from a dockerfile.

- Dockerfile

  A recipe for creating the layers of an image and assembling its content.

- Docker-compose

  Allows you to setup complex systems involving many containers; micro-services environments

- Containers

  Running instances of images. Each container has it is own isolated environment and processes. They are not persistent and volume are necessary if you want to store and keep the data.

- Volumes, and Bind Mounts

  External storage areas that you can mount to your containers to share and persist the data in your containers

- Networks

  Allows containers to communicate with other containers and processes. This is essential for micro-services deployments.

DOCKERFILE

- A text file containing instructions allowing docker to create new images based on the existing one: ***docker build -t name:tag [-f /path/to/dockerfile] .***

Dockerfile example:

```
ARG version=latest  # --build-arg
FROM baseimage:$version
LABEL key=val …
ENV VAR1=VAL1 …
RUN cmd1 && cmd2 && …
COPY src dst
RUN more_commands …
EXPOSE port/protocol
USER …
ENTRYPOINT [“executable”, “param1”, …] #exec form
(OR ENTRYPOINT cmd param1 …) #shell form

CMD [“more param”, …]  #exec form
(OR CMD param1 param2) #shell form
```

- Use RUN to add layers to your image (install new packages on top)
- Use ENTRYPOINT when a command has to be executed when running the image, and use CMD for the parameters that the user may want to change. Otherwise, use CMD.
- RUM, CMD, and ENTRYPOINT come in shell and exec form
  - Shell form → cmd is executable under shell
  - Exec form → cmd is executed without shell
  - Important to know, especially when using environment variables.

HANDS ON

Getting started with Dockers

DOCKER INSTALLATION

- ssh into your instance on the Arbutus cloud
  - ssh -Y -i [YOUR_PRIVATE_KEY] ubuntu@ipaddress
- Update software repository
  - sudo apt update
- Install docker
  - sudo apt install docker.io
- Start the docker engine and enable it to do so as startup
  - sudo systemctl start docker
  - sudo systemctl enable docker
- Add ubuntu user to docker users
  - sudo usermod -aG docker ubuntu
- Check the installation
  - docker info
  - docker --version

DOCKER CHALLENGE 1 – HELLO WORLD!

- Hello world example with dockers!
  - docker run hello-world
- Did you get any output?
- Check whether the container run using ps or ls:
  - docker ps -a
  - docker container ls –a
- What does “docker container ls -aq” do?
- Run the container using start
  - docker start …
- Did you see any output? Check the logs:
  - docker logs …
- Remove the container using rm or prune
  - docker rm …
  - docker container rm …
  - docker container prune
  - docker container ls –a
- Check the current images and remove them
  - docker image ls
  - docker images
  - docker rmi …
  - docker image prune –a
- How to remove all containers without using prune?

DOCKER CHALLENGE 2 – UBUNTU IMAGE

- Search for a ubuntu image
  - *docker search ubuntu*
- Pull the image
- Run it
- Check the status and logs of the container
- Run the container with echo ‘hello world’ command
- Run the container but attach to it
  - docker run -it ubuntu /bin/bash
  - run few commands inside the container

DOCKER CHALLENGE 3 – WEBSITE USING NGINX

- Search for nginx and read how it is deployed
- What to do to map the docker port to a port on the host?
  - Use -p option (or -P with docker port CID PORT)
- Go to http://[your_ip_address]. Do you see anything?
- Use a Dockerfile and change show Hello world instead?
  - mkdir -p nginx/www
  - cd nginx
  - echo ‘<h1>Hello world</h1>’ >> www/index.html
  - #Write Dockerfile and rebuild the image using
  - docker build -t myweb-nginx:v1 .
- Launch your container and see why it is not working
- Fixing, rebuild the image, launch it, and refresh the web page

```
FROM nginx:latest
LABEL maintainer=“xxx@yyy.zz”
COPY www /usr/share/nginx/html/
CMD ["nginx","-d", "daemon off;"]
```

DOCKER CHALLENGE 4 – WEBSITE USING NGINX

- In the previous challenge, does changing www on the local host affects the content of nignx web page?
- What should we do to avoid building the image each time www changes?
  - Think of volumes and bind mounts
- What does the following command do:

  -
    ```
    docker run -d -v /home/ubuntu/winterschool/nginx/www:/usr/share/nginx/html:ro -p 8080:80 --name mynginx_mnt mynginxweb:v1
    ```

DOCKER & SINGULARITY CHALLENGE

- Run python:latest image
- Which version of python it's using?
- Write a python script following the version of python above.
  - The python script file should be called src/mypyapp.py
  - It should print 'Running myapp using ' and the first argument we pass to it
- Write a Dockerfile with a base image python:latest that
  - Copies src/ to /opt/app/bin
  - Adds /opt/app/bin to the path
  - Makes mypyapp.py executable
  - Runs the must command mypyapp.py
  - Supplies a default value to the first argument CMD

- Build the new image and test it
- Creates an image archive for it and upload it to where singularity is installed
- Convert the image to singularity image using
  - singularity build mypyapp.sif docker-archive://mypyapp.tar
- Run the singularity image and check the results
  - On our clusters, module load singularity
  - singularity run ./mypyapp.sif 123 or just ./mypyapp.sif 123

Hints:

```
docker build -t mypyapp:latest ./
docker save mypyapp:latest -o mypyapp.tar
singularity build mypyapp.sif docker-archive://mypyapp.tar
```

QUIZ

- What’s docker?
- Is it the only container technology?
- Can a Windows docker image run on Ubuntu?
- Why docker is becoming very useful in many deployments?
- What are the main components of Docker?
- How to manage each from the command line?
- Which commands allow you to:
  - See the running dockers
  - See all dockers
  - See all images
  - Check the logs of a docker
  - Attach to a docker
- Why should convert my docker to singularity? Is that really necessary?

DOCKER AND SINGULARITY RESOURCES

- https://docs.docker.com/engine/reference/commandline/docker/
- https://sylabs.io/guides/3.5/user-guide/
- https://www.westgrid.ca/events/exploring_containerization_singularity
- https://docs.computecanada.ca/wiki/Singularity
- https://docker-curriculum.com/
- https://github.com/cyberlaboratories/cyberhubs