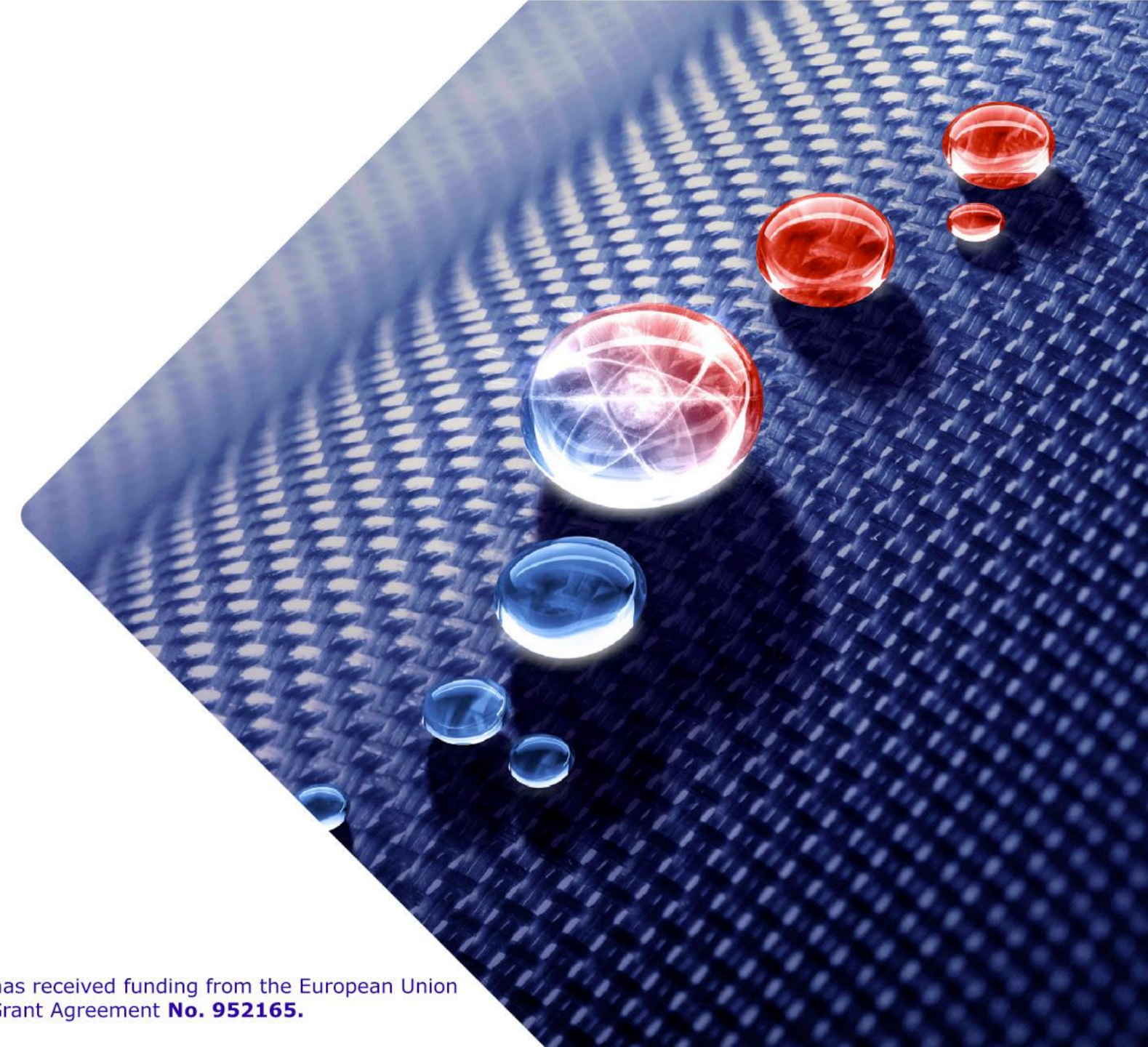
Targeting Real chemical accuracy at the EXascale

Container

Nico Mittenzwey
Kai Löhnig



Targeting Real Chemical Accuracy at the Exascale project has received funding from the European Union Horizon 2020 research and innovation programme under Grant Agreement **No. 952165.**

Why should YOU use Containers?

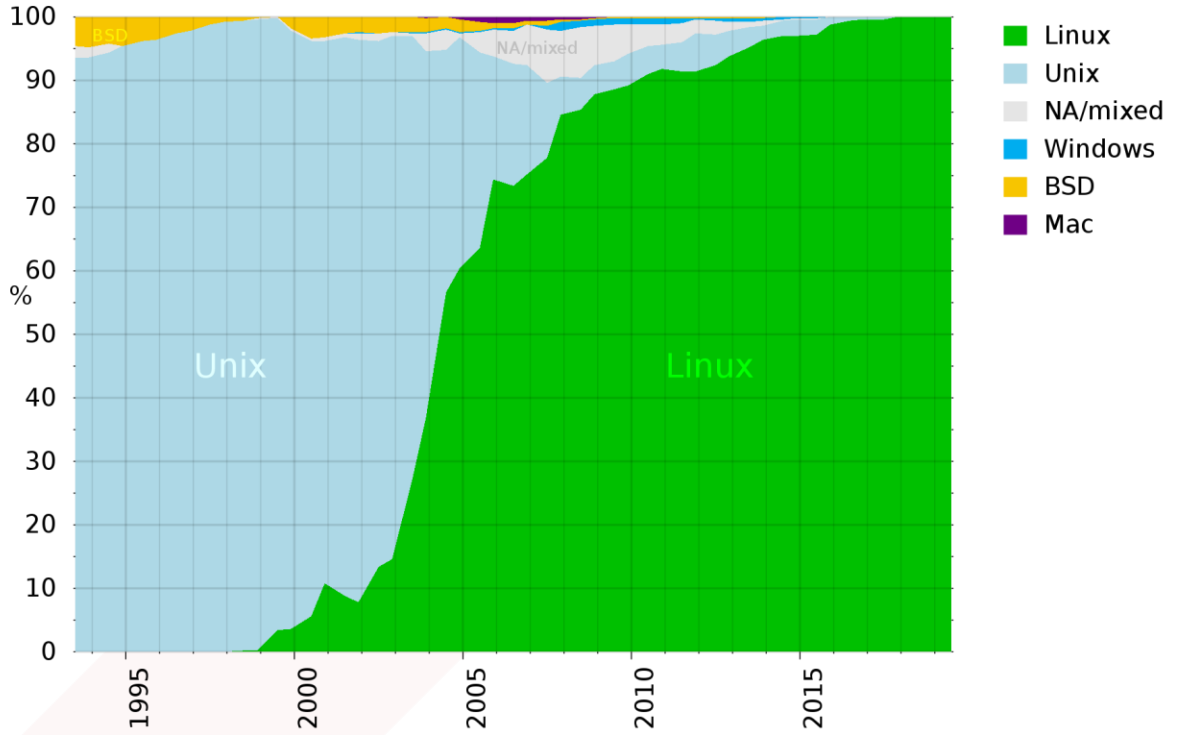
- Users:
  - Ease the setup and running of your applications
  - Ease the portability of your applications
  - Allow reproducibility by others

- Developers:
  - Ease "DevOps"
  - Compile your code for different operation systems and versions

https://unsplash.com/photos/NdFREMS5P08

Rationale

- Your supercomputer most likely uses Linux
- Linux distributions and versions differ
  - RHEL7 vs RHEL8
  - SUSE
  - Ubuntu
- System libraries and their versions differ
  - LibC
  - Math Libraries
  - Python



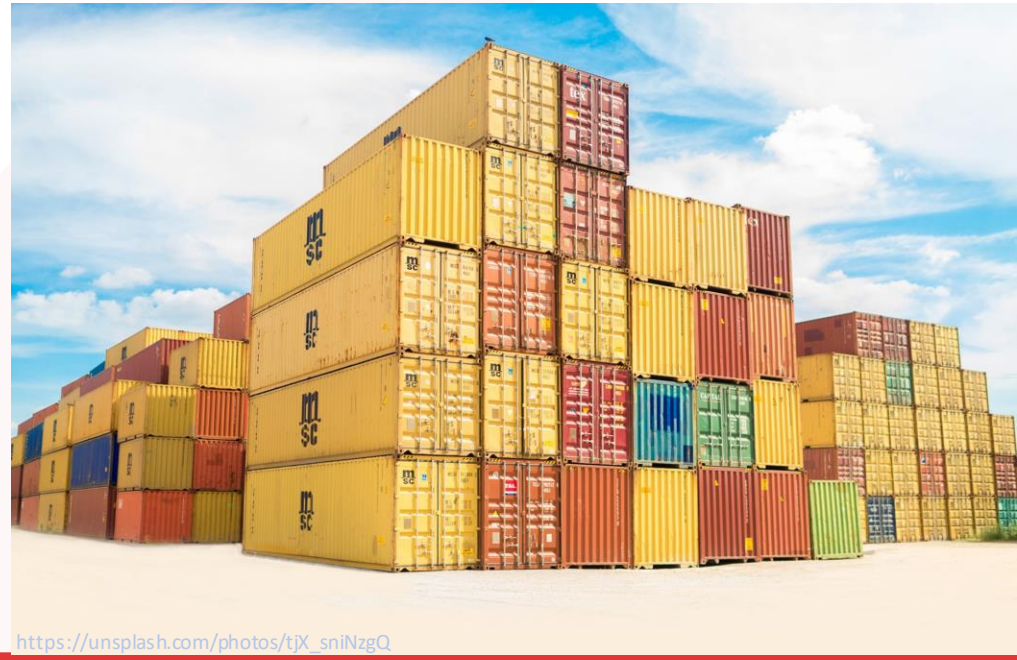
Operating Systems of TOP500 Supercomputers

-> Using and moving applications on multiple systems can be challenging

https://en.wikipedia.org/wiki/Usage_share_of_operating_systems#/media/File:Operating_systems_used_on_top_500_supercomputers.svg

Solution: Have a portable, easily shareable, artifact

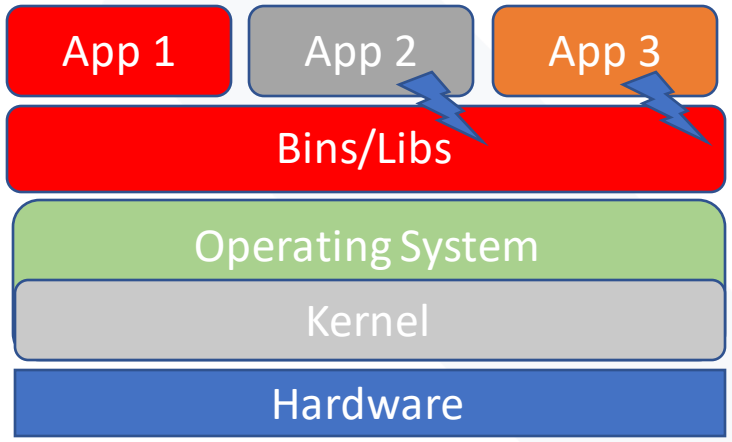
- **Containers** are an intelligent way to package an application
  - Including all needed dependencies
  - Including all needed configurations
  - Isolated environment
  - Shareable either via a repository or as one file
    - https://hub.docker.com/search?q=QMCPACK



https://unsplash.com/photos/tjX_sniNzgQ

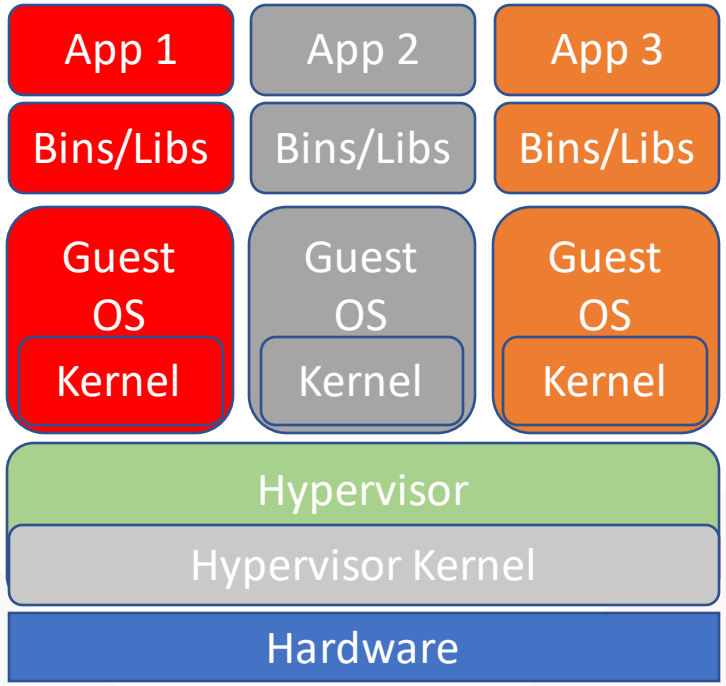
Machines vs. Virtual Maschines vs. Containers

Traditional Machine



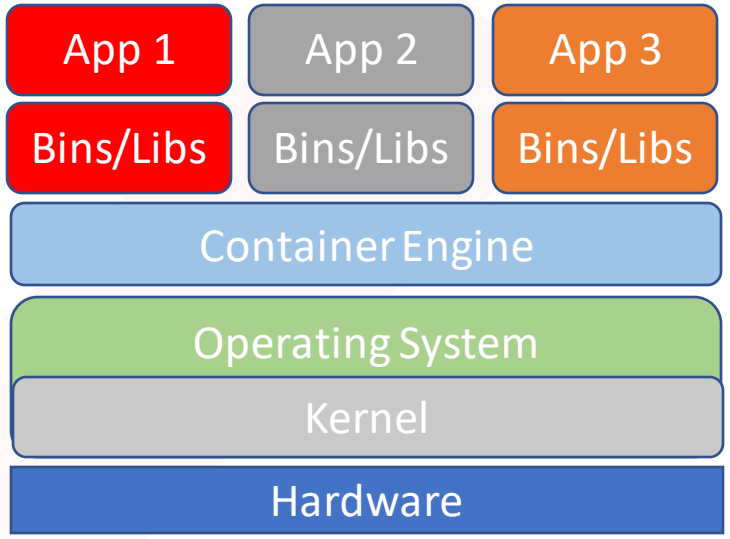
- Pro:
  - Easy to setup
- Cons:
  - Apps may depend on different libs
  - Not portable

Virtual Machines



- Pro:
  - Separation
  - Portable but huge
- Cons:
  - Performance impact
  - Hard to setup

Containers



- Pro:
  - Separation
  - Portable
  - Easy to setup

Container Software Platforms

- Docker (www.docker.com)
  - most used platform with lots of prebuild application containers on https://hub.docker.com/
  - needs root access to install / hard to secure for a multi-user system
  - needs a deamon running on the node

- Pod Manager "Podman" (https://www.podman.io)
  - developed as an alternative to Docker
  - no root access required
  - compatible to Docker command line commands

- Singularity (https://sylabs.io/singularity/)
  - developed as an alternative to Docker for HPC systems
  - no root access required
  - can use Docker containers
  - supports HPC interconnects natively
  - recently introduced a "Pro" version with enterprise support

https://pixabay.com/photos/container-ship-container-ship-port-2786842/

Image

- act as a set of instructions to build a Docker container
- like a class in programming language or blueprint in real world
- contains application code, libraries, tools

Container

- an independet instance of an image
- like an object in programming language or manifactured object from blueprint

Image - Basic Commands

- *docker pull* – downloads images from repository (default hub.docker.com)

```
# docker pull almalinux:8
8: Pulling from library/almalinux
Digest: sha256:b2e1ecb0bac82071625d81bad348cda7dc92b8a21b01a8cc0a6d489e4e0edd2f
Status: Downloaded newer image for almalinux:8
docker.io/library/almalinux:8
```

- *docker image ls* or *docker images* - shows images in local workspace

```
# docker image ls
REPOSITORY              TAG       IMAGE ID       CREATED             SIZE
alma_slurm              latest    06ad41a26a79   About an hour ago   852MB
cw_legacy_webfrontend   latest    82360d76c1a2   3 weeks ago         583MB
almalinux               8         18d68cc33780   4 weeks ago         210MB
debian                  10        730bc7b177ec   6 weeks ago         114MB
```

- *docker image rm* or *docker rmi* - delete images in local workspace

```
# docker rmi almalinux:8
Untagged: almalinux:8
Untagged: almalinux@sha256:b2e1ecb0bac82071625d81bad348cc
```

Image – Example Build/Compile Image

- *build_slurm_el8.dockerfile*

```
ARG BASE_IMAGE

# example: almalinux:8
FROM $BASE_IMAGE

ARG NAME_COMPILE_SCRIPT

USER 0

RUN dnf -y install epel-release dnf-plugins-core && \
    yum config-manager --set-enabled powertools && \
    dnf -y install gcc gcc-c++ perl python3 rpm-build git cmake wget && \
    dnf -y install munge-devel hwloc-devel rrdtool-devel mysql-devel ...

RUN mkdir -p /build/slurm

WORKDIR /build/slurm

# exapmle: my_compile_script.sh
COPY $NAME_COMPILE_SCRIPT /build/slurm

ENTRYPOINT [ "$NAME_COMPILE_SCRIPT", "param1", "valueParam1", "param2", ... ]
```

Image - Basic Commands 2

- *docker build PATH* – builds an image from a dockerfile
- *--file* or *-f* – sets name of dockerfile (default: Dockerfile)
- *--tag* or *-t* – simple tag for the image (default latest)
- *--build-arg* – paramater which can be passed into the container
- Example: *docker build --build-arg "BASIC_IMAGE=almalinux:8" --build-arg "BASIC_IMAGE=my_compile_script.sh" -f ./build_slurm_el8.dockerfile -t alma_slurm .*

Image – Example Build/Compile Image

- my_compile_script.sh

```
#!/bin/bash

wget https://download.schedmd.com/slurm/slurm-${SLURMVERSION}${SLURMRELEASE}.tar.bz2
rpmbuild -ts slurm-${SLURMVERSION}${SLURMRELEASE}.tar.bz2
rpm --install ./SRPMS/slurm-${SLURMVERSION}*.src.rpm
rpmbuild -ba ./SPECS/slurm.spec
```

Container - Basic Commands

- *docker run IMAGE* – creates and executes a container from an image
- *--name* – set name for container
- *--terminal --interactive or -t -i* short *-ti* – command line access inside the container
- *--entrypoint* – overwrite default entrypoint (main routine) of the image
- *--volume or -v* – mounts a directory ("volume") from host into container
- *--rm* – removes container after exit
- *--env or -e* – set environment variables

- Example: *docker run –v /tmp/slurm:/build/slurm -e "SLURMVERSION=20.11.8" --name build_slurm alma_slurm*
- Example debugging: *docker run –ti –entrypoint "/bin/bash" –v /tmp/slurm:/build/slurm --name build_slurm alma_slurm*

Container - Basic Commands

- *docker container ls* - shows running container
- *-a* – shows all container

```
# docker container ls -a
CONTAINER ID   IMAGE                  COMMAND                  CREATED          STATUS
         PORTS                                                                  NAMES
58c6db206833   alma_slurm             "/bin/bash"              49 seconds ago   Exited (0) 46 secon
ds ago                                                                          build_slurm
d7ad61568202   cw_legacy_webfrontend  "/usr/local/share/cl…"   11 days ago      Up 11 days
         0.0.0.0:18000-18004->18000-18004/tcp, :::18000-18004->18000-18004/tcp  cw_legacy_webfronte
nd_container
```

- *docker container rm* or *docker rm* – removes container in local workspace

```
# docker rm build_slurm
build_slurm
```

Container - Basic Commands

- *docker create IMAGE* – creates container from image
- *docker start CONTAINER* – starts stopped or created container
- *docker attach CONTAINER* – connects to running container
- *docker stop CONTAINER* – stops container
- *docker kill CONTAINER* – forces the container to stop

Image – Build Application Image

- Example Folding At Home: https://github.com/linuxserver/docker-foldingathome/blob/master/Dockerfile

```
FROM ghcr.io/linuxserver/baseimage-ubuntu:bionic

# set version label
ARG BUILD_DATE
ARG VERSION
ARG FOLDINGATHOME_RELEASE
LABEL build_version="Linuxserver.io version:- ${VERSION} Build-date:- ${BUILD_DATE}"
LABEL maintainer="aptalca"

#Add needed nvidia environment variables for https://github.com/NVIDIA/nvidia-docker
ENV NVIDIA_DRIVER_CAPABILITIES="compute,video,utility"

# global environment settings
ENV DEBIAN_FRONTEND="noninteractive" \
  MAJOR_VERSION=7.6
```

Image - Build Application Image

```
RUN \
  echo "**** install runtime packages ****" && \
  apt-get update && \
  apt-get install -y \
    jq \
    ocl-icd-libopencl1 && \
  ln -s libOpenCL.so.1 /usr/lib/x86_64-linux-gnu/libOpenCL.so && \
  echo "**** install foldingathome ****" && \
  download_url=$(curl -sL https://download.foldingathome.org/releases.py?series=${MAJOR_VERSION} |
  curl -o \
    /tmp/fah.deb -L \
    ${download_url} && \
  dpkg -x /tmp/fah.deb /app && \
  echo "**** cleanup ****" && \
  apt-get clean && \
  rm -rf \
    /tmp/* \
    /var/lib/apt/lists/* \
    /var/tmp/*

# add local files
COPY root/ /

# ports and volumes
EXPOSE 7396
VOLUME /config
```

Container – Import / Export

- *docker save IMAGE* – saves everything needed to build a container from scratch
- *docker load IMAGE* – loads image from file created with "save" - no hub needed

- *docker export CONTAINER* – export a container including its current file system into a file
- *docker import CONTAINER* – imports container including its file system from file

Questions?

Thank you!

Follow us

company/trex-eu

@trex_eu

Targeting Real Chemical Accuracy at the Exascale project has received funding from the European Union Horizon 2020 research and innovation programme under Grant Agreement **No. 952165.**

TREX
Targeting Real chemical accuracy at the EXascale