14th AI RoundTable

# Getting started with Docker

May 28, 2023

12:00PM-13:30PM (EST)

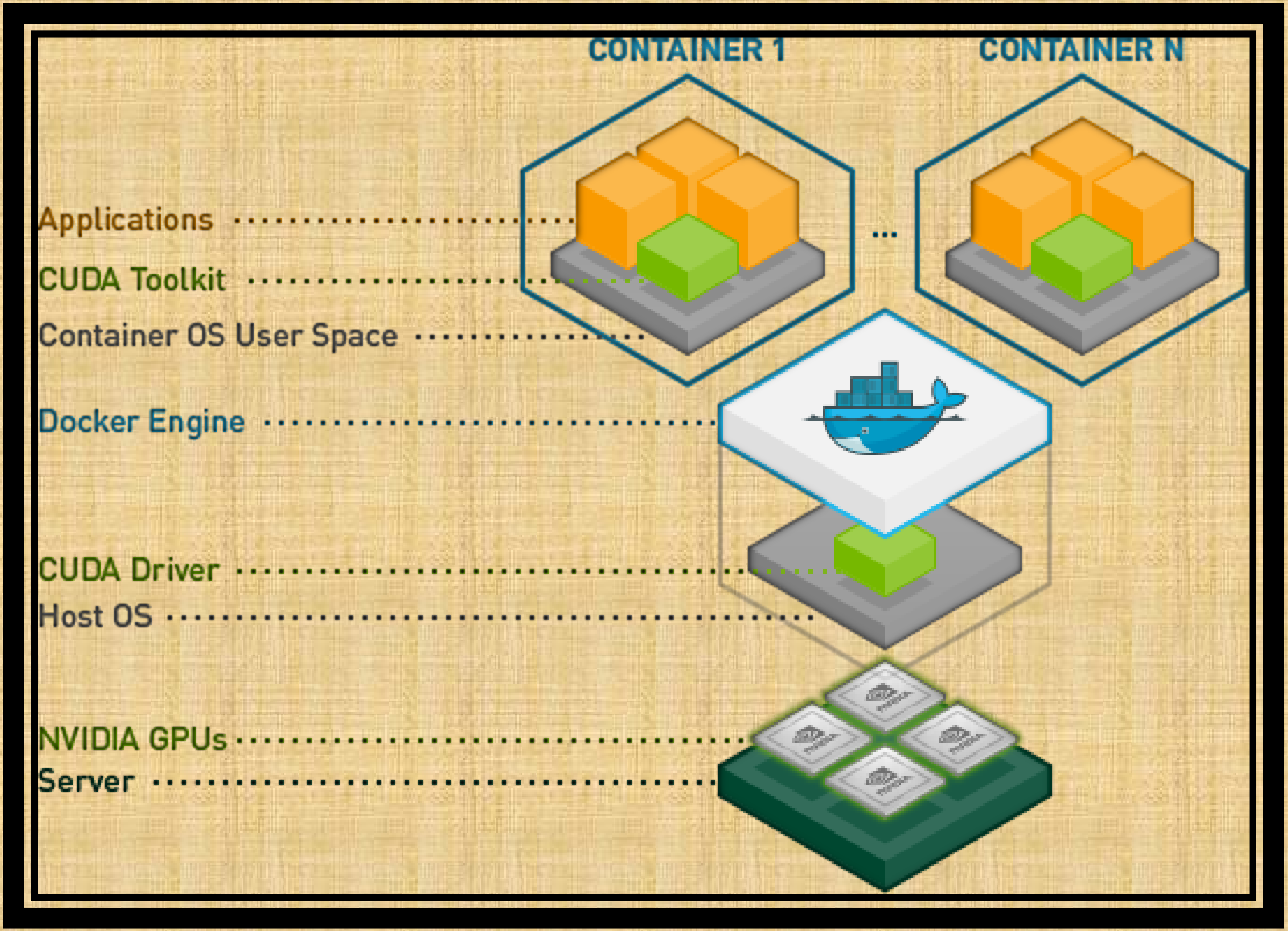
With: Hamed Farkhari

docker Ubuntu NVIDIA

## Outlines

- What is Docker?
- Install Nvidia Driver, CUDA
- Install Docker
- Images and Containers

## What is Docker?



## First, Backup your images before reinstall docker

### Images

```
sudo docker images
```

### Save to tar file

```
docker   save   -o   <path for generated tar file>    <image name:tag>
sudo   docker   save   -o   ~/Documents/xxx.tar   xxx:latest
```

### After reinstalling docker, load tar file

```
docker   load   -i   <path to tar file>
```

Requirements

## Remove old Docker

https://docs.docker.com/engine/install/ubuntu/

## It will remove all images (backup your images)

```
sudo  apt-get   remove  docker   docker-engine  docker.io  containerd  runc
sudo  apt-get   purge     docker-ce   docker-ce-cli   containerd.io
sudo  rm   -rf   /var/lib/docker
sudo  rm   -rf   /var/lib/containerd
sudo  apt-get    purge   docker*
sudo  apt-get  --purge   remove   docker*

sudo   apt-get   autoremove
sudo   apt-get   autoclean
```

Install Nvidia Driver, CUDA

Requirements

What is Docker?

## Remove Nvidia

```
sudo   /usr/bin/nvidia-uninstall
sudo   apt-get      purge   nvidia-*
sudo   apt-get   --purge   remove   nvidia-*   "*nvidia*"
```

## Remove CUDA Toolkit

```
sudo apt-get --purge remove  "*cuda*"   "*cublas*"   "*cufft*"   "*cufile*"
"*curand*"   "*cusolver*"   "*cusparse*"   "*gds-tools*"   "*npp*"
"*nvjpeg*"   "nsight* "

sudo   apt-get   autoremove
sudo   apt-get   autoclean
```

## Check CUDA directories and delete them

```
cd       /usr/local
sudo   rm   -rdf   cuda-12.1   cuda
ls
```

Install Nvidia Driver, CUDA

Requirements

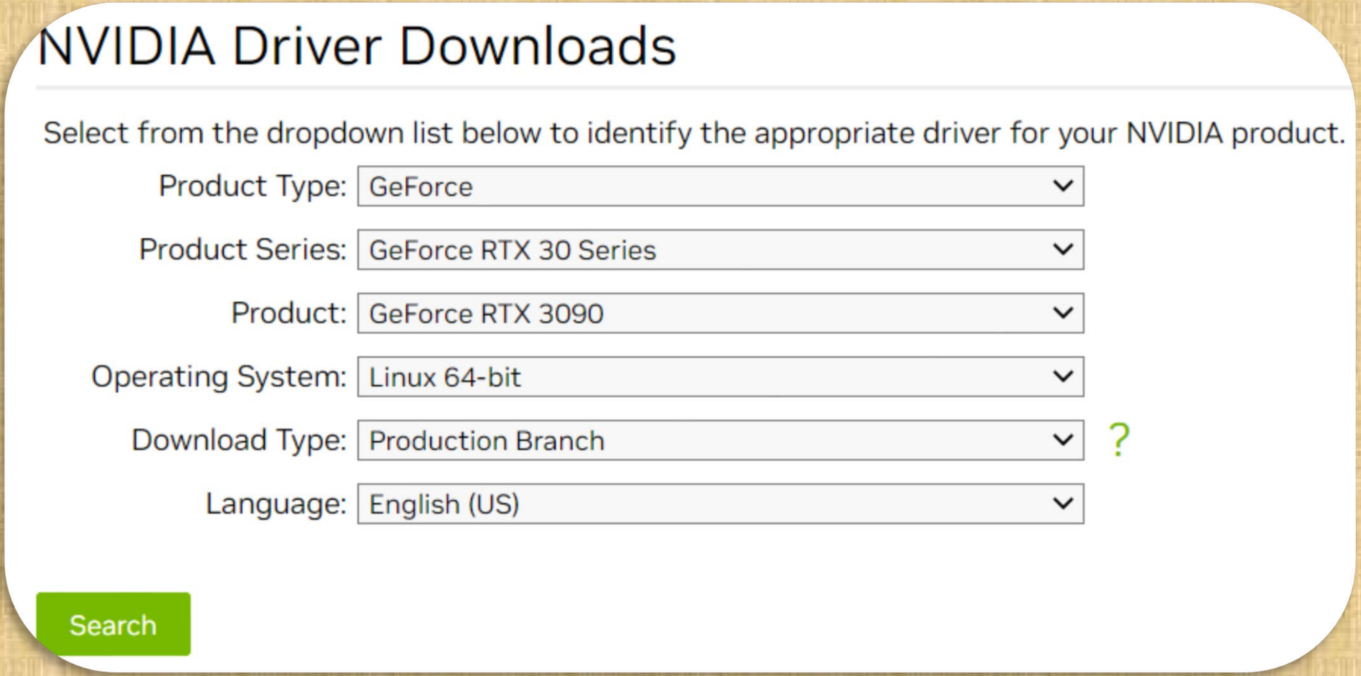
What is Docker?

## Install Requirements

```
sudo   apt   update
sudo   apt-get   update

sudo  apt  install  --upgrade  build-essential  libglvnd-dev  pkg-config
sudo  apt-get  install  zlib1g
sudo  apt-get  install  g++   freeglut3-dev   libx11-dev   libxmu-dev
sudo  apt-get  install  libxi-dev   libglu1-mesa   libglu1-mesa-dev
sudo  apt-get  install  libfreeimage3   libfreeimage-dev

gcc  --version
g++  --version
ldd  --version

sudo   apt   upgrade
sudo   apt   update
```

## Download Nvidia Driver

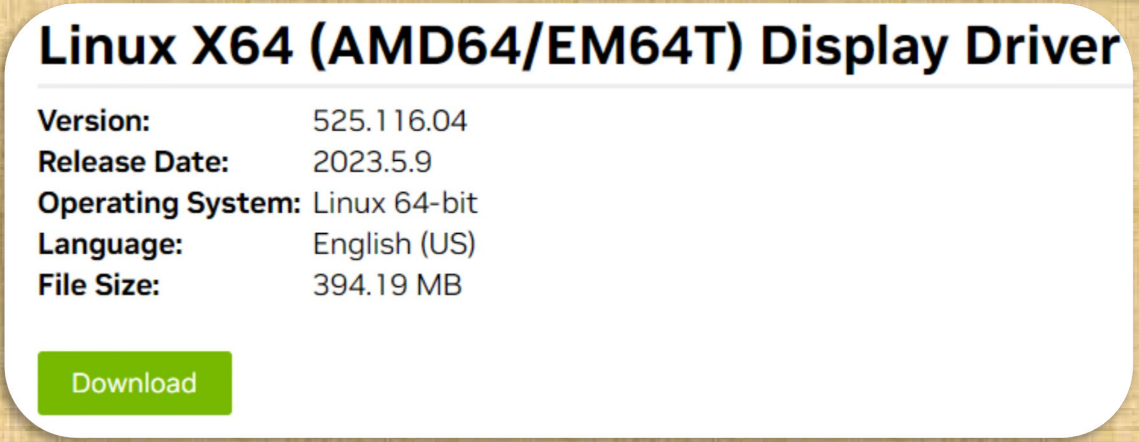
http://www.nvidia.com/Download/index.aspx



NVIDIA Driver Downloads

Select from the dropdown list below to identify the appropriate driver for your NVIDIA product.

| | |
|---|---|
| Product Type: | GeForce |
| Product Series: | GeForce RTX 30 Series |
| Product: | GeForce RTX 3090 |
| Operating System: | Linux 64-bit |
| Download Type: | Production Branch |
| Language: | English (US) |



Linux X64 (AMD64/EM64T) Display Driver

| | |
|---|---|
| **Version:** | 525.116.04 |
| **Release Date:** | 2023.5.9 |
| **Operating System:** | Linux 64-bit |
| **Language:** | English (US) |
| **File Size:** | 394.19 MB |

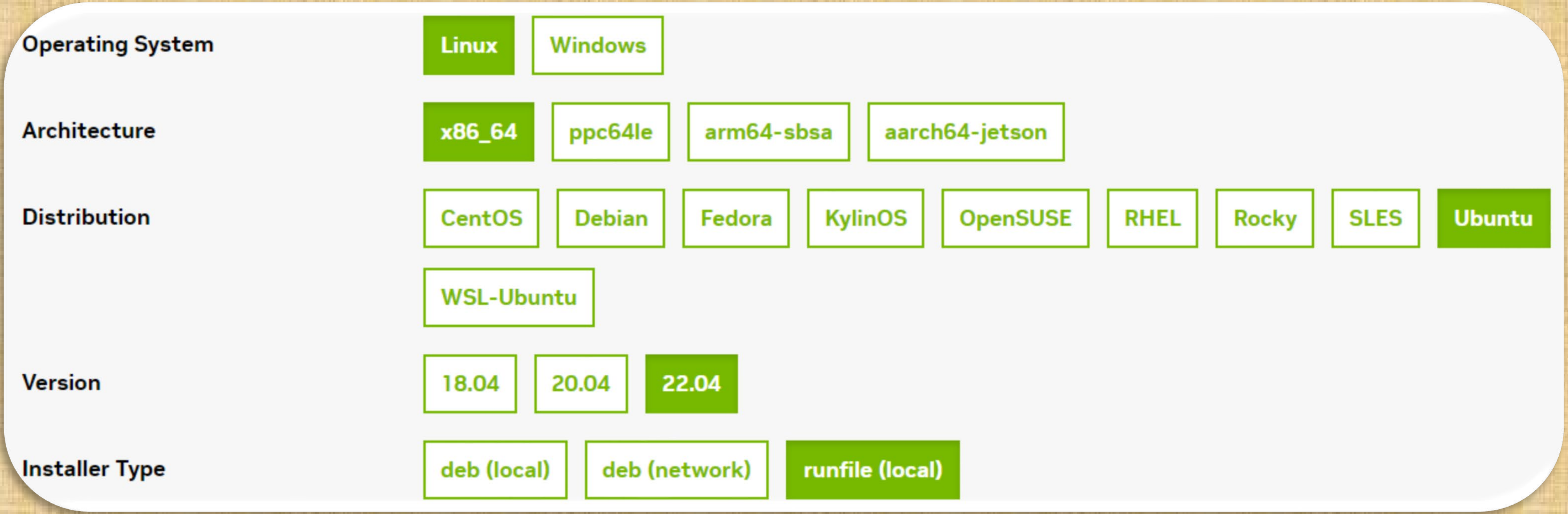
Install Docker

Install Nvidia Driver, CUDA

What is Docker?

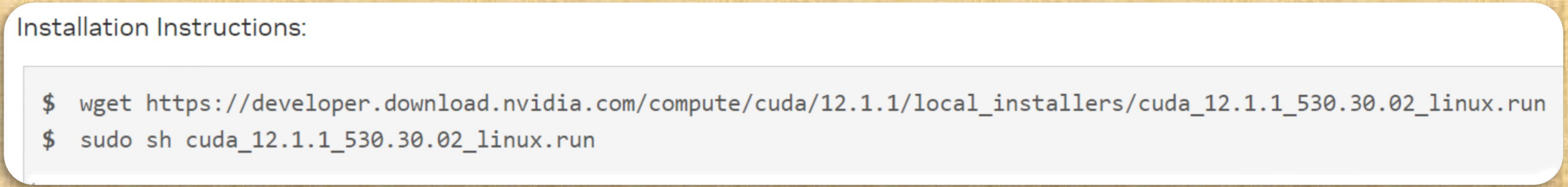
## Download CUDA

https://developer.nvidia.com/cuda-downloads



| | |
|---|---|
| Operating System | [✓ Linux] Windows |
| Architecture | [✓ x86_64] ppc64le arm64-sbsa aarch64-jetson |
| Distribution | CentOS Debian Fedora KylinOS OpenSUSE RHEL Rocky SLES [✓ Ubuntu] WSL-Ubuntu |
| Version | 18.04 20.04 [✓ 22.04] |
| Installer Type | deb (local) deb (network) [✓ runfile (local)] |

```
cd \
cd   Downloads/
```



Installation Instructions:

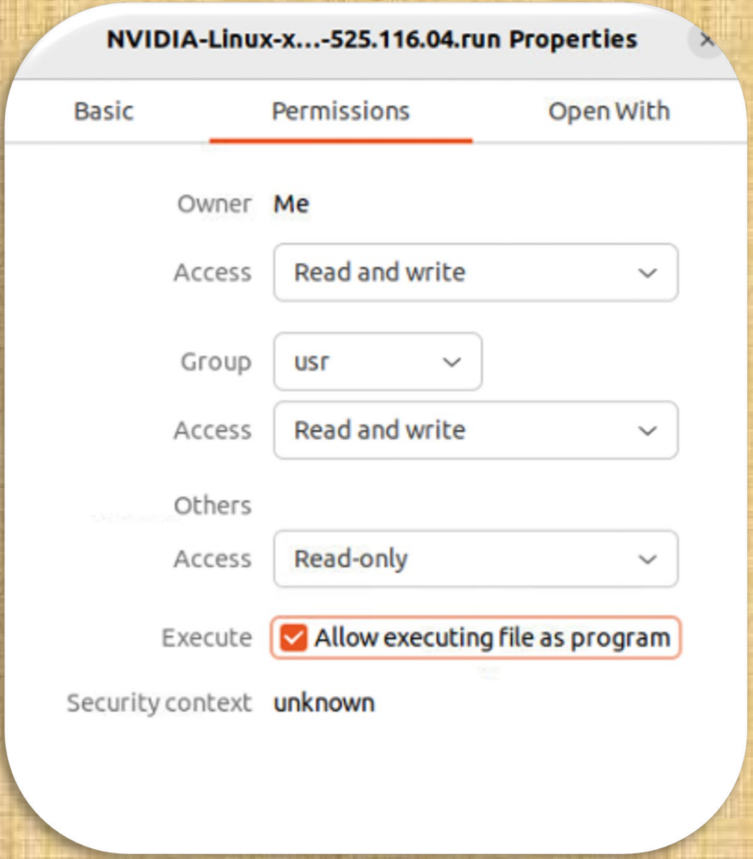
```
$  wget https://developer.download.nvidia.com/compute/cuda/12.1.1/local_installers/cuda_12.1.1_530.30.02_linux.run
$  sudo sh cuda_12.1.1_530.30.02_linux.run
```

Install Nvidia Driver, CUDA

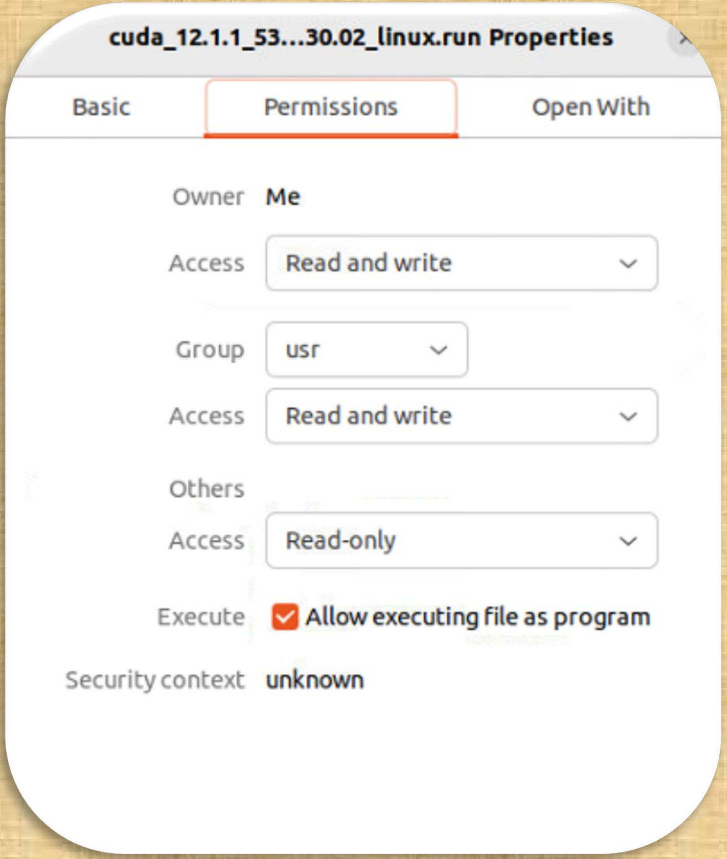
What is Docker?

## Execute Permission

```
chmod   +x    NVIDIA-Linux-x86_64-525.116.04.run
chmod   +x    cuda_12.1.1_530.30.02_linux.run
```


NVIDIA-Linux-x...-525.116.04.run Properties

| | |
|---|---|
| Owner | Me |
| Access | Read and write |
| Group | usr |
| Access | Read and write |
| Others | |
| Access | Read-only |
| Execute | [✓ Allow executing file as program] |
| Security context | unknown |


cuda_12.1.1_53...30.02_linux.run Properties

| | |
|---|---|
| Owner | Me |
| Access | Read and write |
| Group | usr |
| Access | Read and write |
| Others | |
| Access | Read-only |
| Execute | [✓ Allow executing file as program] |
| Security context | unknown |

Install Docker

Install Nvidia Driver, CUDA

What is Docker?

## Disable Nouveau Nvidia driver

https://linuxconfig.org/how-to-disable-blacklist-nouveau-nvidia-driver-on-ubuntu-20-04-focal-fossa-linux

```
sudo   bash   -c   "echo blacklist nouveau > /etc/modprobe.d/blacklist-nvidia-nouveau.conf "
sudo   bash   -c   "echo options nouveau modeset=0 >> /etc/modprobe.d/blacklist-nvidia-nouveau.conf "
```

## Confirm

```
cat   /etc/modprobe.d/blacklist-nvidia-nouveau.conf

# blacklist nouveau
# options nouveau modeset=0

sudo  update-initramfs   -u
sudo  reboot
```

## Stop Desktop Manager (Ctrl+Alt+F1)

```
sudo  telinit  3
```

## Install Nvidia Driver, CUDA

```
cd \
cd  Downloads/
```

**Install Nvidia driver**

```
sudo  bash  NVIDIA-Linux-x86_64-525.116.04.run
```

**Install CUDA**

```
sudo  bash  cuda_12.1.1_530.30.02_linux.run
```

**Restart**

```
sudo  apt-get  update
sudo  reboot
```

Install Docker

Install Nvidia Driver, CUDA

What is Docker?

## After Install CUDA

Its important to recognize cuda 12.1, not cuda 10

**Please make sure that**

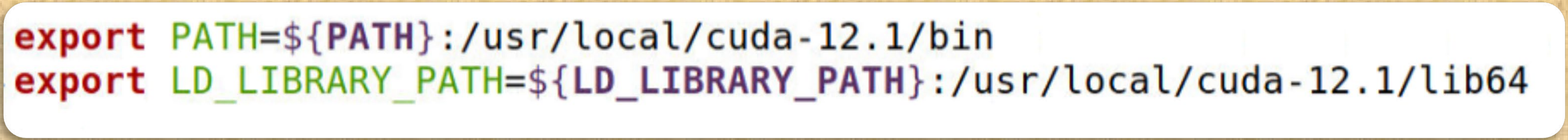
- PATH includes /usr/local/cuda-12.1/bin
- LD_LIBRARY_PATH includes /usr/local/cuda-12.1/lib64, or, add /usr/local/cuda-12.1/lib64 to /etc/ld.so.conf and run ldconfig as root

## Add path

```
gedit   ~/.bashrc
```

## Add the following variables

```
export PATH=/usr/local/cuda-12.1/bin${PATH:+:${PATH}}
export LD_LIBRARY_PATH=/usr/local/cuda-12.1/lib64${LD_LIBRARY_PATH:+:${LD_LIBRARY_PATH}}

export PATH=${PATH}:/usr/local/cuda-12.1/bin
export LD_LIBRARY_PATH=${LD_LIBRARY_PATH}:/usr/local/cuda-12.1/lib64
```



```
export PATH=${PATH}:/usr/local/cuda-12.1/bin
export LD_LIBRARY_PATH=${LD_LIBRARY_PATH}:/usr/local/cuda-12.1/lib64
```

## Save and exit

```
sudo    ldconfig
source  ~/.bashrc
nvcc  -V
```

## Output should be like this

```
nvcc: NVIDIA (R) Cuda compiler driver
Copyright (c) 2005-2023 NVIDIA Corporation
Built on Mon_Apr__3_17:16:06_PDT_2023
Cuda compilation tools, release 12.1, V12.1.105
Build cuda_12.1.r12.1/compiler.32688072_0
```

Images and

Containers

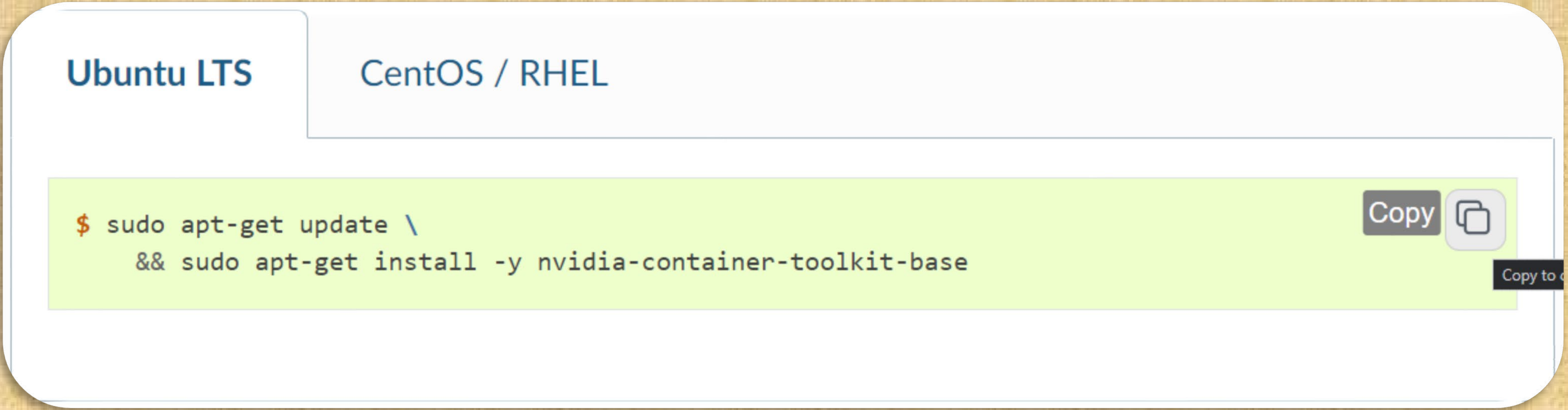
Install Docker

Install Nvidia Driver, CUDA

## Install Docker

https://docs.nvidia.com/datacenter/cloud-native/container-toolkit/install-guide.html#pre-requisites

## Pre-Requisites
## Container Device Interface (CDI) Support



```
$ sudo apt-get update \
    && sudo apt-get install -y nvidia-container-toolkit-base
```

Images and

Containers

Install Docker

Install Nvidia Driver, CUDA

## Install Docker

https://docs.nvidia.com/datacenter/cloud-native/container-toolkit/install-guide.html#pre-requisites

1. Pre-Requisites
2. Container Device Interface (CDI) Support
3. Docker
   - Installing on Ubuntu and Debian
   - Setting up Docker
4. Setting up NVIDIA Container Toolkit

```
sudo docker run --rm --runtime=nvidia --gpus all nvidia/cuda:11.6.2-base-ubuntu20.04 nvidia-smi
```

5. Containerd
6. Step 1: Install Containerd

```
sudo ctr image pull docker.io/library/hello-world:latest \
    && sudo ctr run --rm -t docker.io/library/hello-world:latest hello-world
```

Images and Containers

## Download Latest TensorFlow

https://catalog.ngc.nvidia.com/orgs/nvidia/containers/tensorflow

## Download Latest PyTorch

https://catalog.ngc.nvidia.com/orgs/nvidia/containers/pytorch

## Docker Hub

https://hub.docker.com

```
docker  image  pull   <image_name:tag>
```

https://docs.docker.com/engine/reference/commandline/pull/

Images and Containers

Install Docker

## Docker Run

```
docker run --gpus all -it --rm  nvcr.io/nvidia/tensorflow:xx.xx-tfx-py3
```

```
docker run --gpus all --ipc=host --ulimit memlock=-1 --ulimit stack=67108864 -p 8888:8888 -it --rm
-v  $(realpath ~/Desktop/workspace):/workspace/ubuntu-desktop    nvcr.io/nvidia/pytorch:23.05-py3
jupyter notebook    --ip  0.0.0.0   --no-browser    --allow-root
```

```
docker run --gpus all --ipc=host --ulimit memlock=-1 --ulimit stack=67108864 -p 8888:8888 -it --rm
-v   $(realpath ~/Desktop/workspace):/workspace/ubuntu-desktop
nvcr.io/nvidia/tensorflow:23.05-tf2-py3    jupyter notebook   --ip 0.0.0.0   --no-browser   --allow-root
```

Images and Containers

Install Docker

## <Customization> Modifying a Container Image

## Docker Images/Containers

```
sudo   docker   images
sudo   docker   container   ls  -a
```

## Run Container

```
docker   run   --gpus   all   -it   --rm  nvcr.io/nvidia/tensorflow:23.04-tf2-py3
```

## Obtain Container id

```
sudo   docker   ps   --no-trunc
```

## Container Shell / Install new app in shell

```
sudo   docker   exec   -it   630b26d0971e   bash
```

## Save new image with new name

```
sudo   docker   commit   <container_id>     <image_name:tag>
sudo   docker   commit   630b26d0971e     tensorflow_new:new_tag
```

Images and Containers

Install Docker

## Install New app in Shell

```
apt-get   update
pip       install   --upgrade   pip
python  -m   pip   install   --upgrade   pip

pip       install   seaborn
```

## Remove Containers

```
sudo   docker   container   rm   <container-id>
```

## Remove Images

```
sudo   docker   image   rm   [OPTIONS] IMAGE [IMAGE...]
sudo   docker   image   rm  tensorflow/my_tensorflow_v2
sudo   docker   rmi      <image-tag>
```

Thanks

Any Question ?