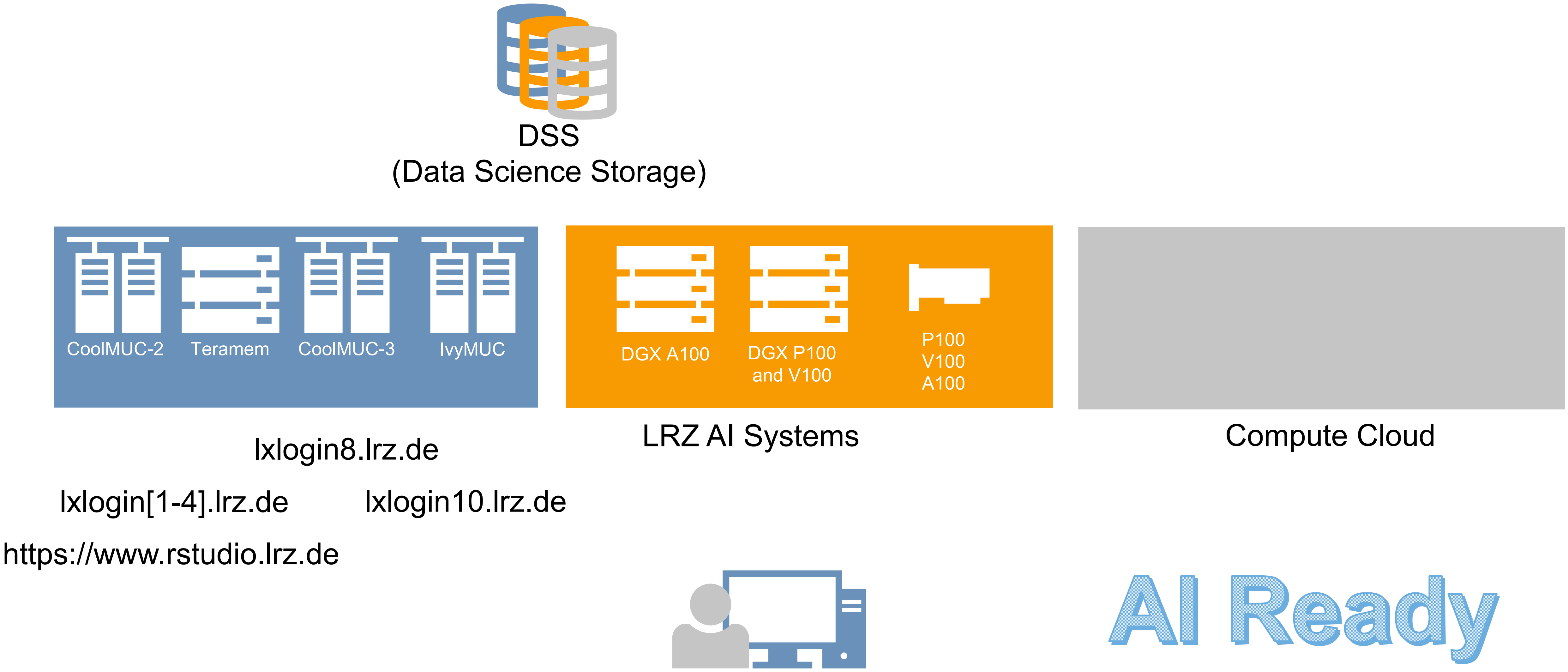
lrz

Leibniz Supercomputing Centre

of the Bavarian Academy of Sciences and Humanities

Deep Learning on the LRZ AI Systems

An High Level Overview of Some LRZ Resources



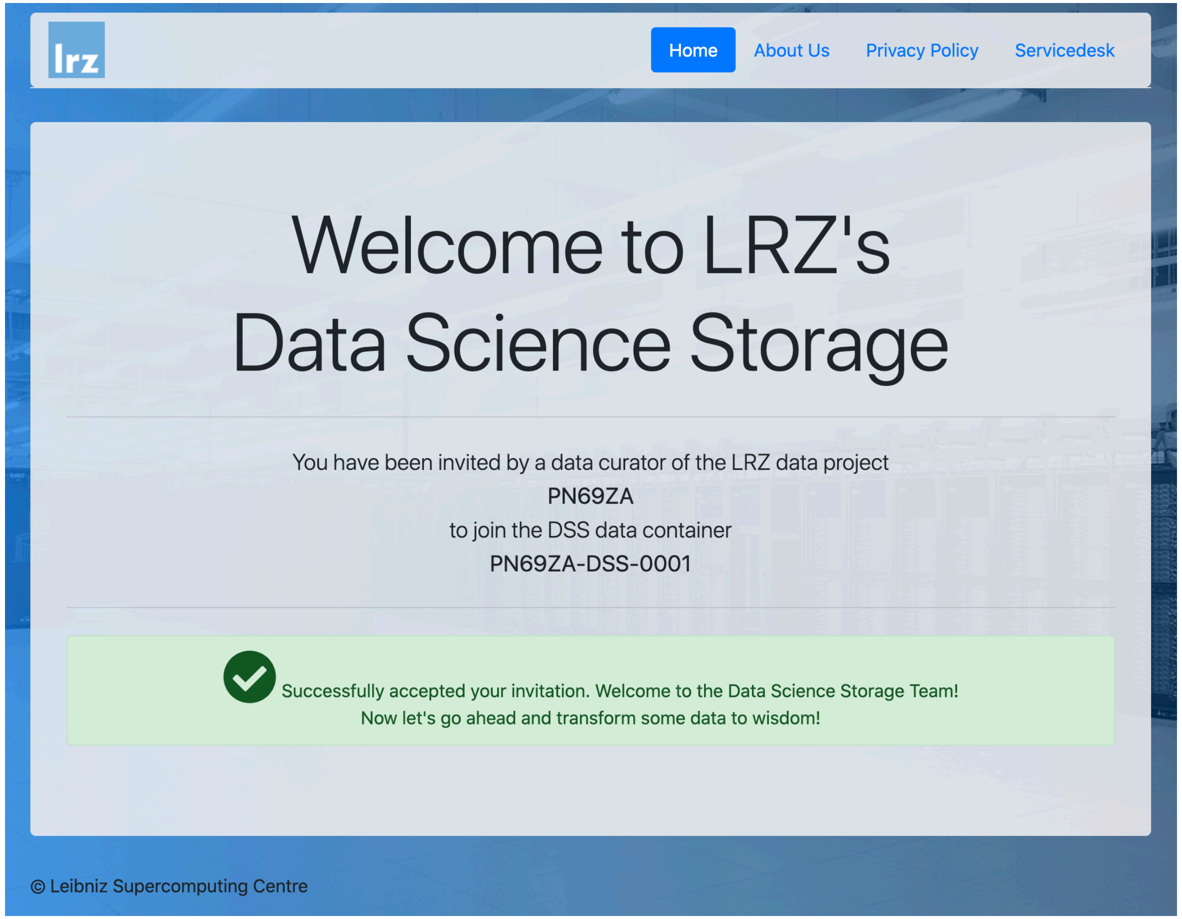
The LRZ AI System

Who can access?

- User requirements to get the access:
  1. Own a Linux Cluster account,
  2. Send a request through a service request ticket explaining the intended use.

- Upon approval, you will be invited to a DSS container - You need to accept this invitation before being able to access the AI systems!

- This DSS container will be used as your $HOME (although this is going to change in the future.)



Resources Overview

	Slurm Partition	Number of nodes	CPUs per node	Memory per node	GPUs per node	Memory per GPU
DGX A100 Architecture	lrz-dgx-a100-80x8	4	256	2 TB	8 NVIDIA A100	80 GB
	lrz-dgx-a100-40x8	1	256	1 TB	8 NVIDIA A100	40 GB
DGX-1 V100 Architecture	lrz-dgx-1-v100x8	1	80	512 GB	8 NVIDIA Tesla V100	16 GB
DGX-1 P100 Architecture	lrz-dgx-1-p100x8	1	80	512 GB	8 NVIDIA Tesla P100	16 GB
HPE Intel Skylake + NVIDIA Node	lrz-hpe-p100x4	1	64	256 GB	4 NVIDIA Tesla P100	16 GB
V100 GPU Nodes	lrz-v100x2 (default)	4	20	368 GB	2 NVIDIA Tesla V100	16 GB

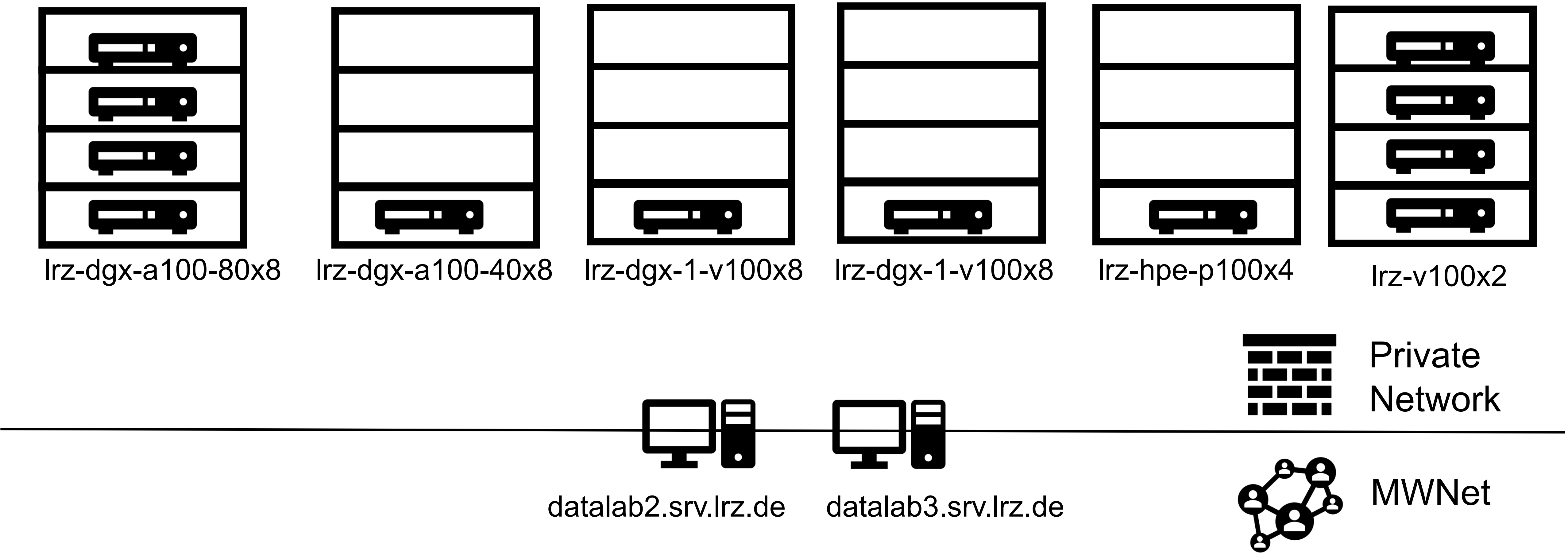
Using the Cluster

- SLURM: Simple Linux Utility for Resource Management
  - open source
  - fault-tolerant
  - highly scalable

- Cluster management and job scheduling

- (Three) main tasks
  - allocates exclusive and/or non-exclusive access to resources (compute nodes)
  - provides a framework for starting, executing, and monitoring work on the allocated nodes
  - arbitrates contention for resources by managing a queue of pending work.

LRZ AI System Configuration



Accessing the LRZ System

- Login node datalab2.srv.lrz.de accessible via ssh

```
ssh -Y datalab2.srv.lrz.de -l xxyyyzz
```

- From the login node, jobs are submitted to the hardware described at the beginning of this course using SLURM
- A couple of handy SLURM commands

```
$ squeue
```

```
$ sinfo
```

```
$ salloc / scancel
```

```
$ srun
```

Allocating and Starting Jobs Interactively

- Get resources allocated

```
$ salloc -p dgx --ntasks=8 --gres=gpu:8
```

Indicate that access to 8 GPU resources is required

resource queue

start n tasks in the machine per node (one per GPU required by some software framework)

- Submit start job in the allocated resources

```
$ srun --pty bash
```

```
$ srun hostname
```

Allocating and Starting Jobs in Batch Mode

- Batch jobs are the preferred way of using the LRZ AI Systems.
- The **sbatch** command submits jobs described in a *sbatch script* file.
- Two additional arguments required in sbatch scripts: *output* and *error messages* file.
- After the preamble, the job to be executed is described.

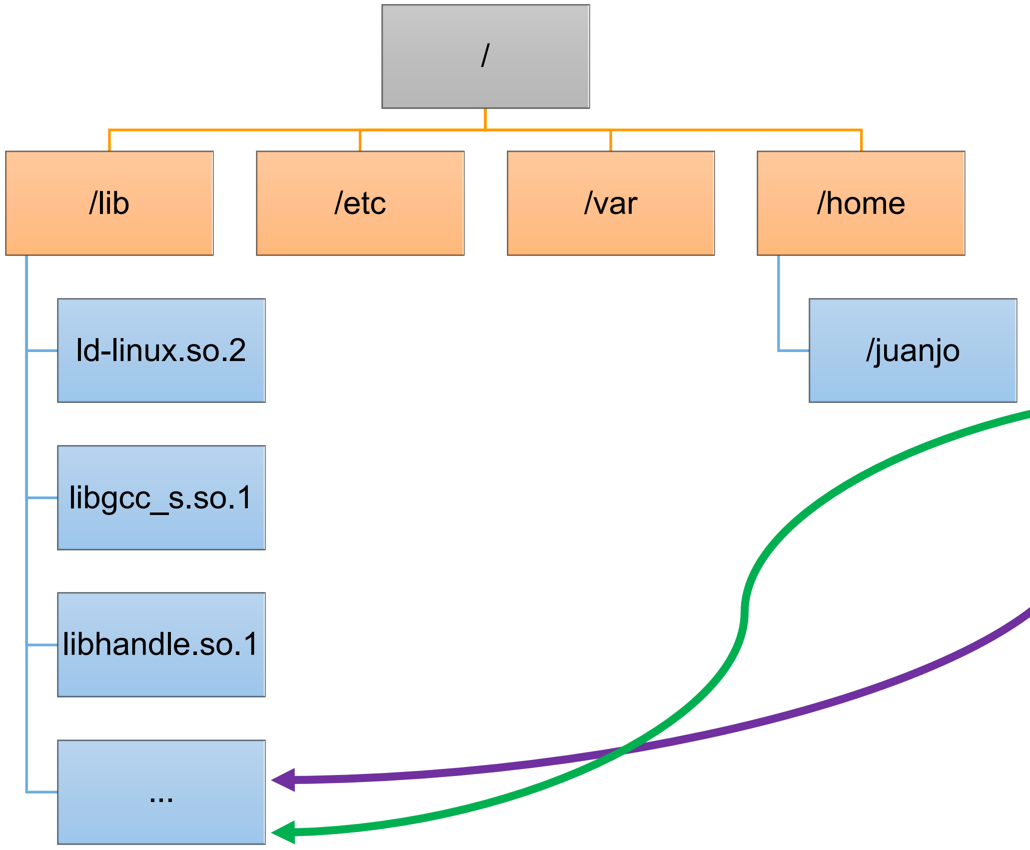
```
#!/bin/bash
#SBATCH -N 1
#SBATCH -p dgx-1-p100
#SBATCH --gres=gpu:2
#SBATCH --ntasks=2

srun nvidia-smi
```

```
$ sbatch test.sbatch
```

User Defined Software Stack: Container Technologies

Typical Linux File System

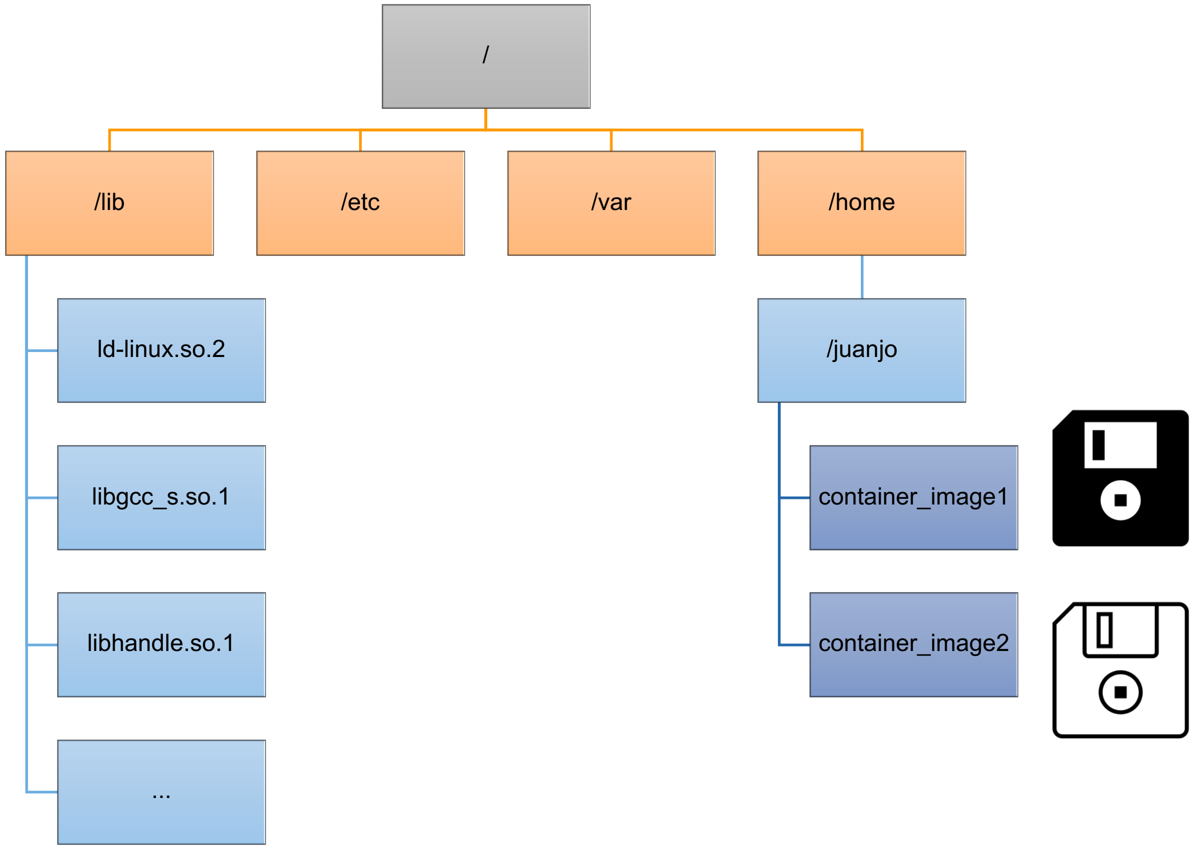


Program In Execution in Linux

```
$ ldd /bin/program
linux-vdso.so.1 (0x00007fff204e8000)
libselinux.so.1 => /lib/x86_64-linux-gnu/libselinux.so.1 (0x00007f411cc6a000)
libc.so.6 => /lib/x86_64-linux-gnu/libc.so.6 (0x00007f411ca78000)
libpcre2-8.so.0 => /usr/lib/x86_64-linux-gnu/libpcre2-8.so.0 (0x00007f411c9e8000)
libdl.so.2 => /lib/x86_64-linux-gnu/libdl.so.2 (0x00007f411c9e2000)
/lib64/ld-linux-x86-64.so.2 (0x00007f411ccd0000)
libpthread.so.0 => /lib/x86_64-linux-gnu/libpthread.so.0 (0x00007f411c9bf000)
```

User Defined Software Stack: Container Images

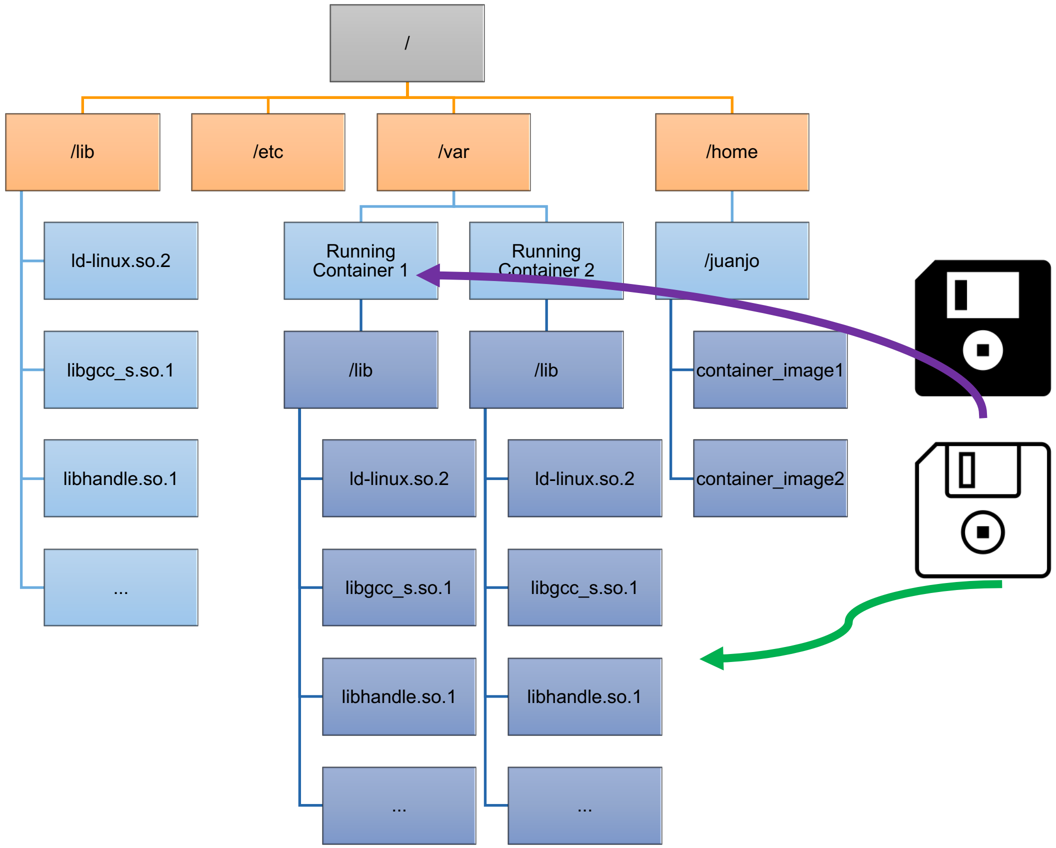
Typical Linux File System



- Typically, a single compressed file
  - It contains a complete Linux File System + Metadata
- Different container technologies might:
  - use different formats
    - e.g., OCI format is **a specification for container images based on the Docker Image Manifest Version 2, Schema 2 format**
  - hide images to users
- Are meant to be static
- Not to be confused with a docker file

User Defined Software Stack: Container

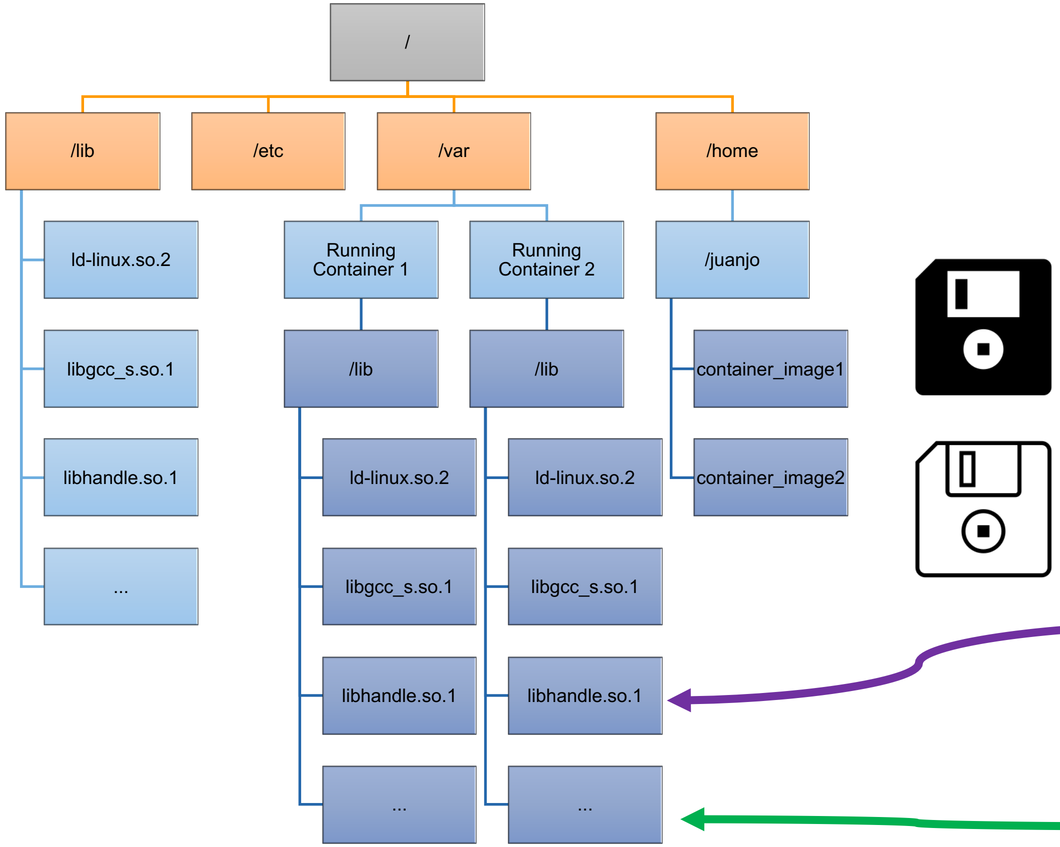
Typical Linux File System



- A running instance of a container image
  - A complete Linux File System within a Linux File System
    - Libraries might be different (versions)
    - Provided programs might be different
- Specific program in charge of unpacking the image and storing it within the proper folder
  - Docker, Podman, Enroot, etc.
- More than one container can
  - Exist at any point in time
  - Be generated from a single image

Understanding Containers: Container

Typical Linux File System



- It is possible to “run a process within a container”
  - Confine the process to the content of the container File System
  - Specific program in charge for confining and running the process within the container
    - docker run/start, enroot start

```
$ ldd /bin/program
linux-vdso.so.1 (0x00007fff204e8000)
libselinux.so.1 => /lib/x86_64-linux-gnu/libselinux.so.1
(0x00007f411cc6a000)
libc.so.6 => /lib/x86_64-linux-gnu/libc.so.6
(0x00007f411ca78000)
libpcre2-8.so.0 => /usr/lib/x86_64-linux-gnu/libpcre2-8.so.0
(0x00007f411c9e8000)
libdl.so.2 => /lib/x86_64-linux-gnu/libdl.so.2
(0x00007f411c9e2000)
/lib64/ld-linux-x86-64.so.2 (0x00007f411ccd0000)
libpthread.so.0 => /lib/x86_64-linux-gnu/libpthread.so.0
(0x00007f411c9bf000)
```

Process run within Running Container 2

The Enroot Container Technology

- Containerized applications with `enroot`, a rootless container runtime by Nvidia
- Slightly different workflow than with `Docker`

#1 Import a container from a valid repository

```
$ enroot import docker://ubuntu
```

#2 Create an enroot container

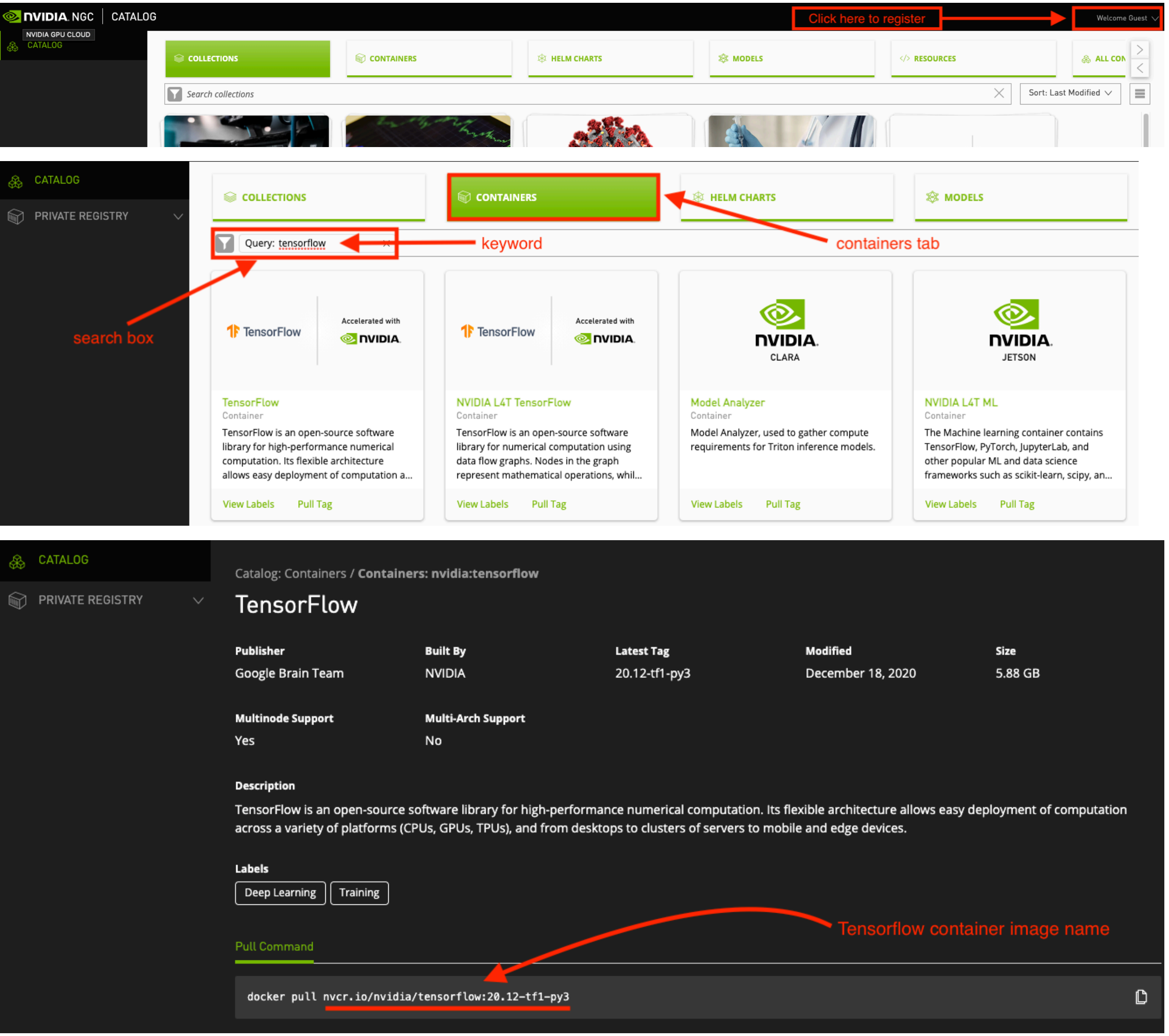
```
$ enroot create ubuntu.sqsh
```

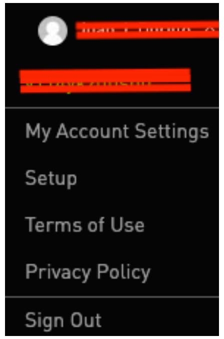
#3 Start/run an application within the container

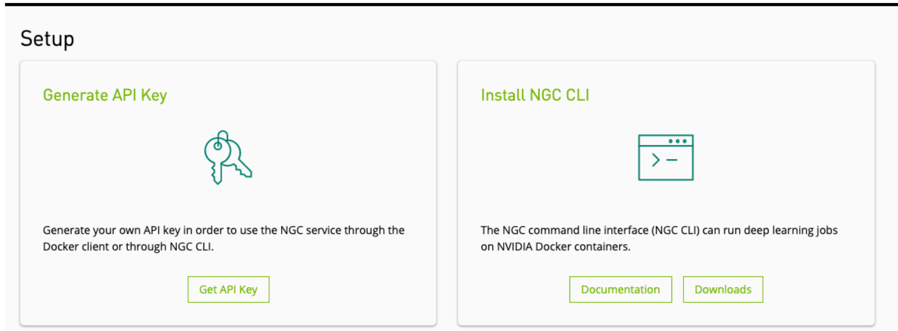
```
$ enroot start ubuntu
```

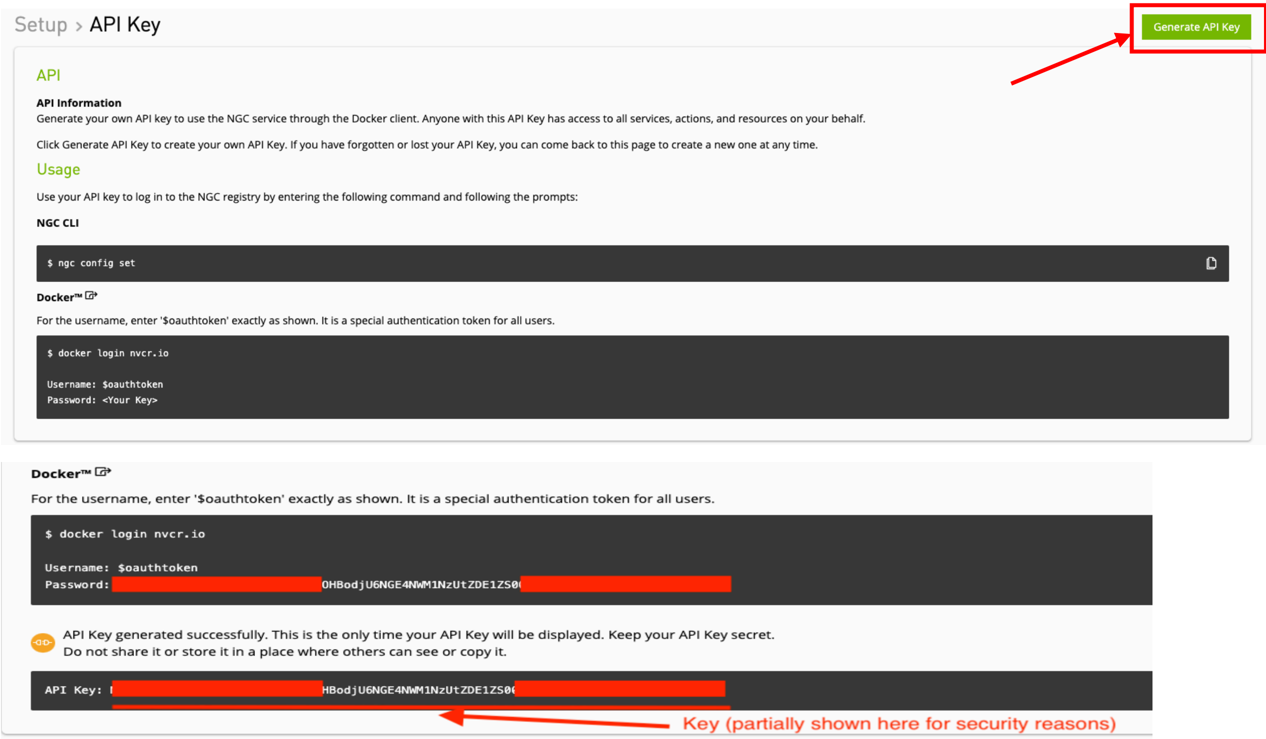
- It should be noticed than the workflow in the AI System consist in submitting jobs that run containerized within an enroot defined container

Nvidia NGC: Container Images Repository









Beyond Existing NGC: Creating Custom Enroot Image

- Get resources / get an allocation of resources
  - salloc -p test-v100x2 -q testing --gres=gpu:1
- Open a terminal on the allocated resources
  - srun --pty bash
- (Optional) Create a base container image (e.g., pulling from NGC)
  - enroot import -o pytorch_base.sqsh docker://nvcr.io#nvidia/pytorch:22.06-py3
- Create a container out of an image
  - enroot create --name pytorch_container pytorch_base.sqsh

Beyond Existing NGC: Creating Custom Enroot Image

- Start a terminal your recently created container
  - enroot start pytorch_container bash
- Modify your container accordingly
- Leave your container
- Export the created container as an image
  - enroot export --output hugging_face.sqsh pytorch_container

Using the Cluster with Enroot Containers

- Get resources allocated

```
$ salloc -p dgx --ntasks=8 --gres=gpu:8
```

Indicate that access to the GPU resources is required

- Submit containerized job

mounting outside folders inside the container

the container previously created

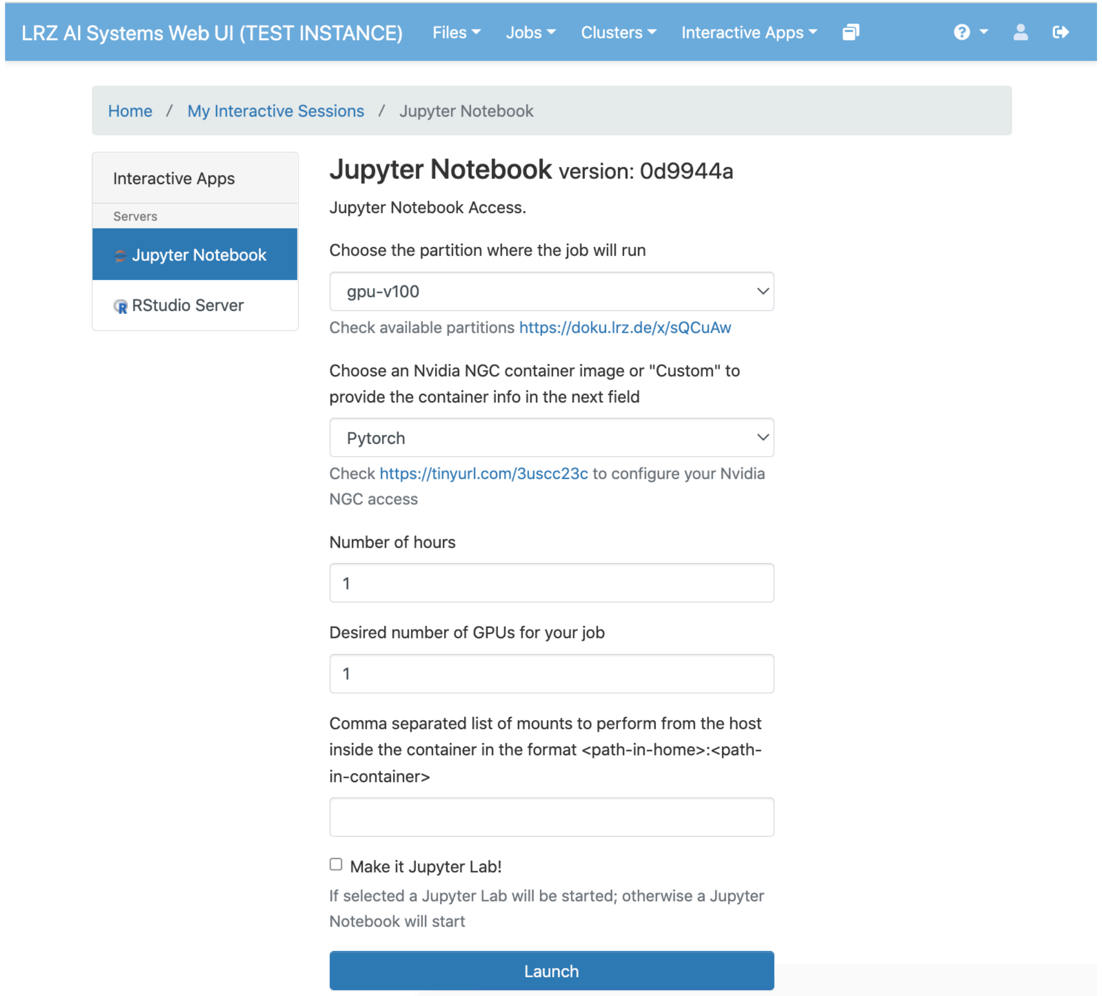
```
$ srun --pty enroot start --mount ./data-test:/mnt/data-test ubuntu bash
```

- Meet the pyxis plugin: container creating and job submission in a single step

```
$ srun --container-mounts=./data-test:/mnt/data-test --container-name=horovod
--container-image='horovod/horovod+0.16.4-tf1.12.0-torch1.1.0-mxnet1.4.1-
py3.5' bash
```

Open On Demand: Web Frontend for the LRZ AI System

- Interactive web service for AI systems where Jupyter Notebook, JupyterLab and RStudio Server environments are available, at https://datalab3.srv.lrz.de.
- Given that requested resources are available, the status of the session will change from "Queued" to "Starting" and finally "Running".
- https://doku.lrz.de/display/PUBLIC/LRZ+AI+Systems

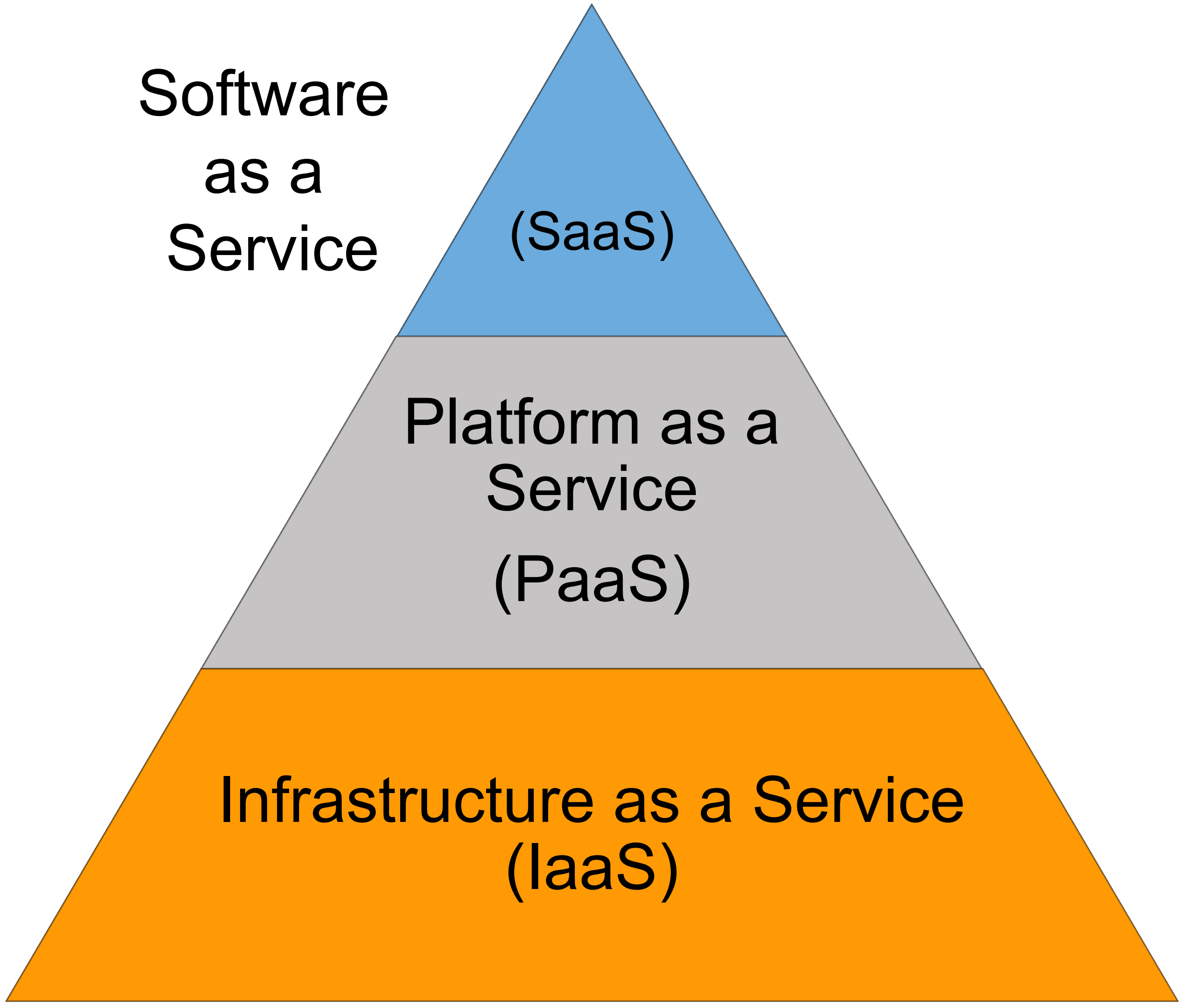


The Compute Cloud

Cloud Computing Characteristics

Description	Characteristic
A consumer can request and receive access to a service offering without an administrator or some sort of support staff having to fulfil the request manually.	on demand
Cloud services should be easy to access. Ideally only a basic network connection should be required.	broad network access
Ability to grow with user demand. If the system is well defined it should be relatively easy for the provider to add more	flexible
Cloud services must have the ability to measured usage. Usage can be quantified using various metrics, such as time, bandwidth used, and data used. This ability to measure allow what is known as pay as you go model.	measurable
A user will not need all the resources available to her. When resources are not used, they should be released, and other users can benefit of it or they can be simply not used (not consuming energy.)	resource pooling

Cloud Services



- SaaS – Fully developed software solution to be used
  - e.g., Google Drive
- PaaS – Provides a framework on top of which is possible to build, deploy, and manage software products
  - e.g., Heroku
- IaaS - Provides a completely virtualized computing infrastructure provisioned and managed over the internet
  - e.g., LRZ Compute Cloud

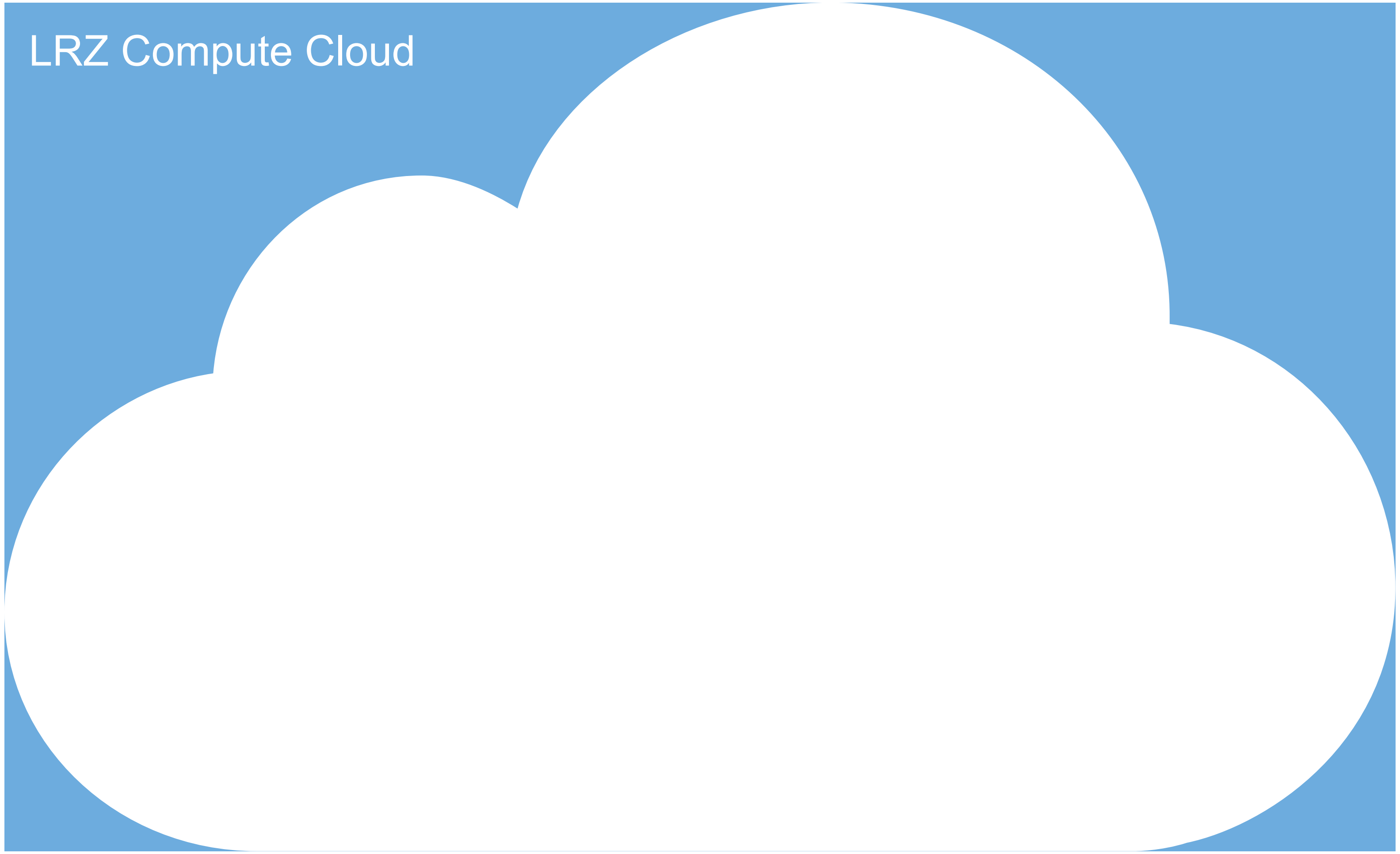
OpenStack: The Engine of the LRZ Compute Cloud

- What do we need for transforming a set of resources (data center) into a cloud?
  - to manage/admin the hardware
  - to provision machines to users
  - to allow users to authenticate
  - to manage the network across resources
  - …

> OpenStack is a cloud operating system that controls large pools of compute, storage, and networking resources throughout a datacenter, all managed and provisioned through APIs with common

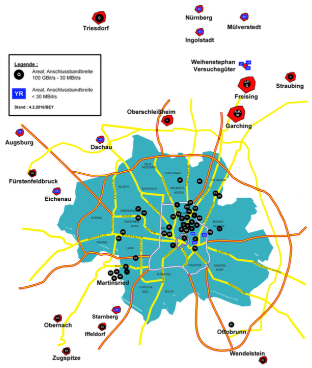
- OpenStack bundles together a bunch of different technologies, addressing the different needs transforming resources into a Cloud Service

The LRZ Compute Cloud at a Glance



Internet

MWN



OpenStack - Terminology

- Image

A single file which contains a virtual disk with a bootable operating system installed on it. Images are like a template of a computer's root drive. They contain the operating system and can also include software and layers of your application, such as database servers, web servers, and so on.

OpenStack - Terminology

- Instance

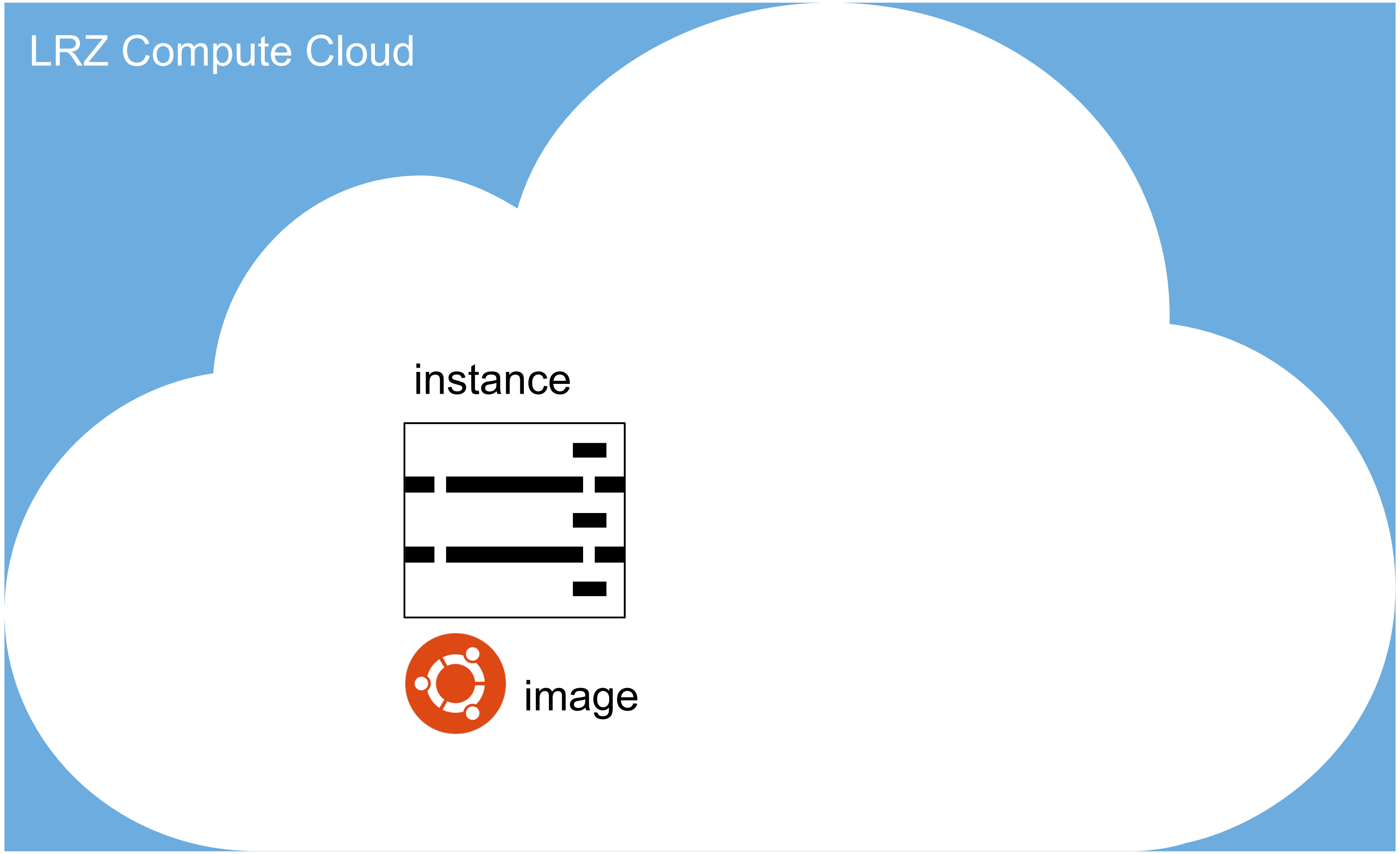
  A copy of an image running as a virtual server the cloud. We will also call it server.

- Flavor

  Flavors define the compute, memory, and storage capacity of instances. To put it simply, a flavor is an available hardware configuration for a server.

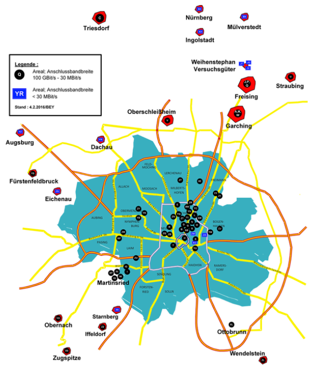
Name	vCPUs	RAM	Remarks	Access
tiny	1	512 MB	for testing purposes only, most Operating Systems will not boot due to restricted resources	public
nvidia-v100.2	40	700 GiB	use 2 GPUs on a GPU node (use entire GPU node)	restricted, contact us
nvidia-v100.1	20	350 GiB	use 1 GPU on a GPU node	restricted, contact us
lrz.xlarge	10	47.5 GiB	use 1/4 compute node	public
lrz.xhuge	48	1488 GiB	use 1/4 of the hugemem node	restricted, contact us
lrz.small	1	4.75 GiB	use 1/40 compute node	public
lrz.medium	2	9.5 GiB	use 1/20 compute node	public
lrz.large	4	19 GiB	use 1/10 compute node	public
lrz.huge	24	744 GiB	use 1/8 of the hugemem node	restricted, contact us
lrz.4xlarge	40	190 GiB	use entire compute node	restricted, contact us
lrz.4xhuge	192	5952 GiB	use entire hugemem node	restricted, contact us
lrz.2xlarge	20	95 GiB	use 1/2 compute node	restricted, contact us
lrz.2xhuge	96	2976 GiB	use 1/2 of the hugemem node	restricted, contact us

The LRZ Compute Cloud at a Glance



Internet

MWN

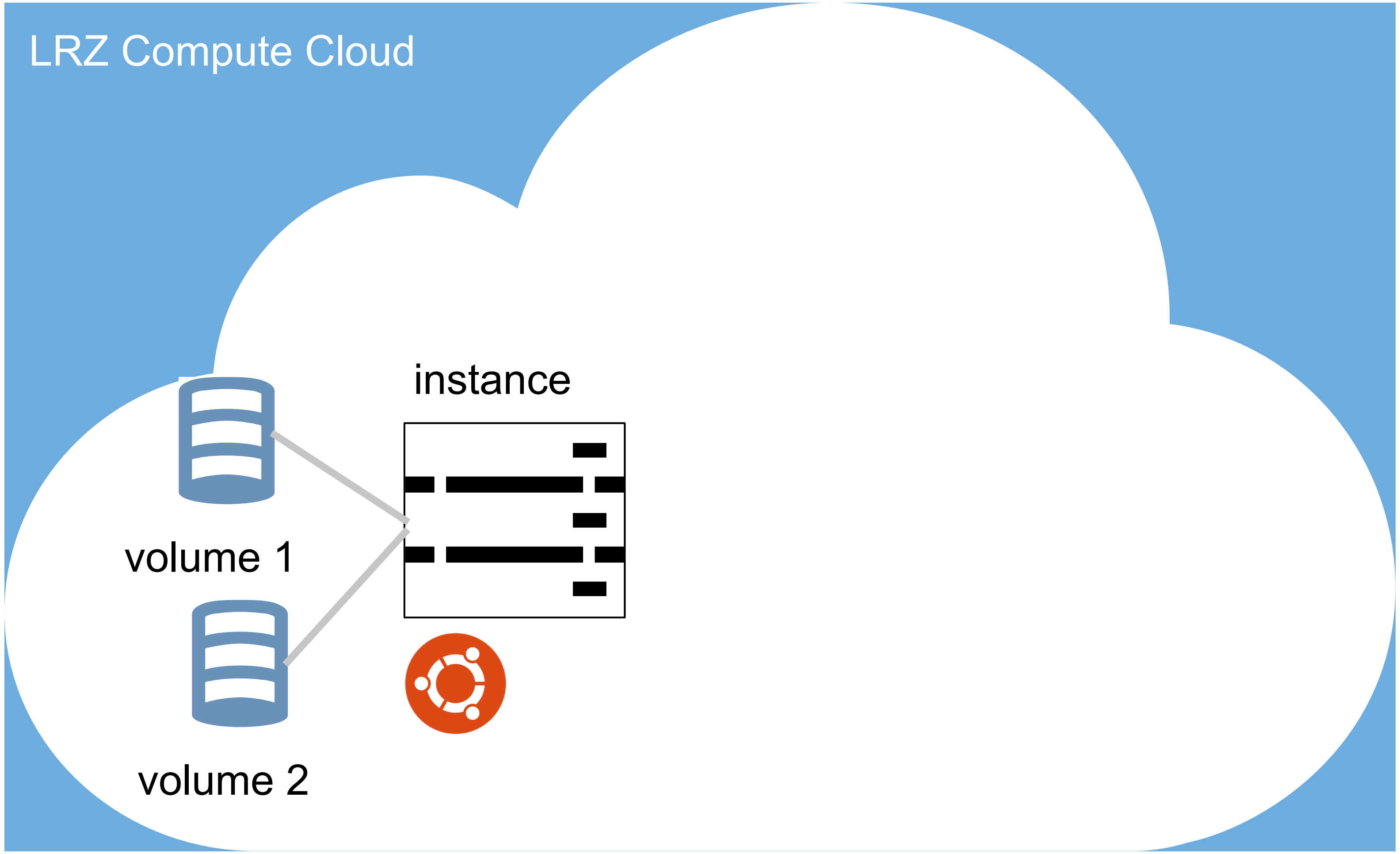


OpenStack - Terminology

- Volume

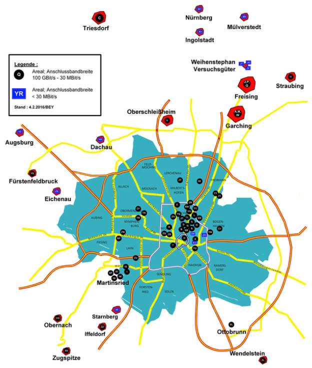
A volume is a detachable block storage device, similar to a USB hard drive. You can attach a volume to only one instance. But an instance can attach several volumes

The LRZ Compute Cloud at a Glance



Internet

MWN



OpenStack - Terminology

- Networking

OpenStack provides networks, subnets, and routers as object abstractions. Each abstraction has functionality that mimics its physical counterpart: networks contain subnets, and routers route traffic between different subnets and networks. Instances are created within internal private networks. These networks can be routed to external networks (e.g., Internet or or MWN) via a virtual router.

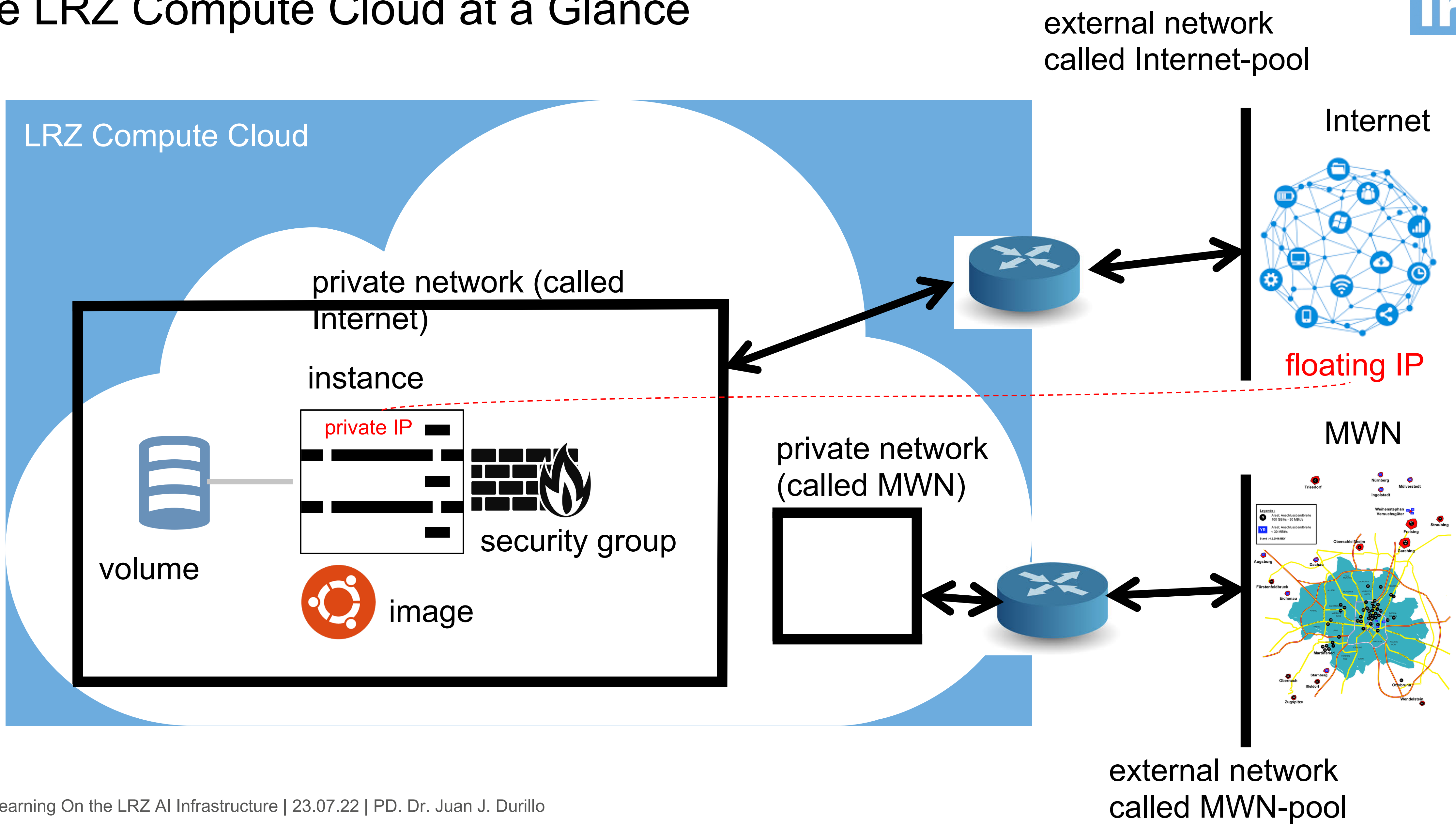
- Private and Floating IP

Each instance has a fixed IP within its private Network. That IP can be associated to an IP of the external network that network is connected by means of what it is called *floating IP address*. The floating IP address will allow addressing the instance from the outside.

- Security group

A security group acts as a virtual firewall for servers and other resources on a network. It is a container for rules for allowing different types of network traffic to and from an instance.

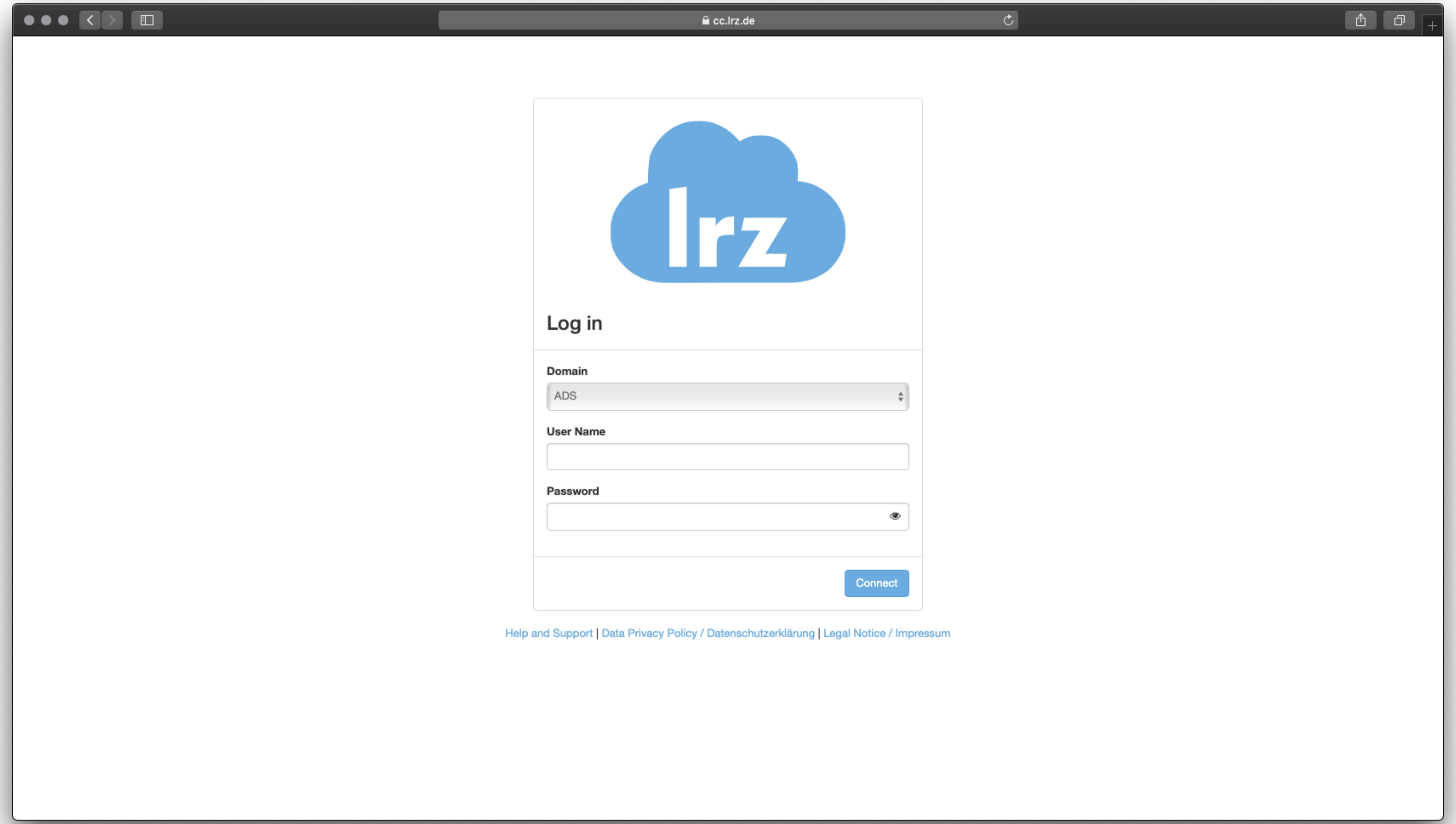
The LRZ Compute Cloud at a Glance



A server on the Compute Cloud

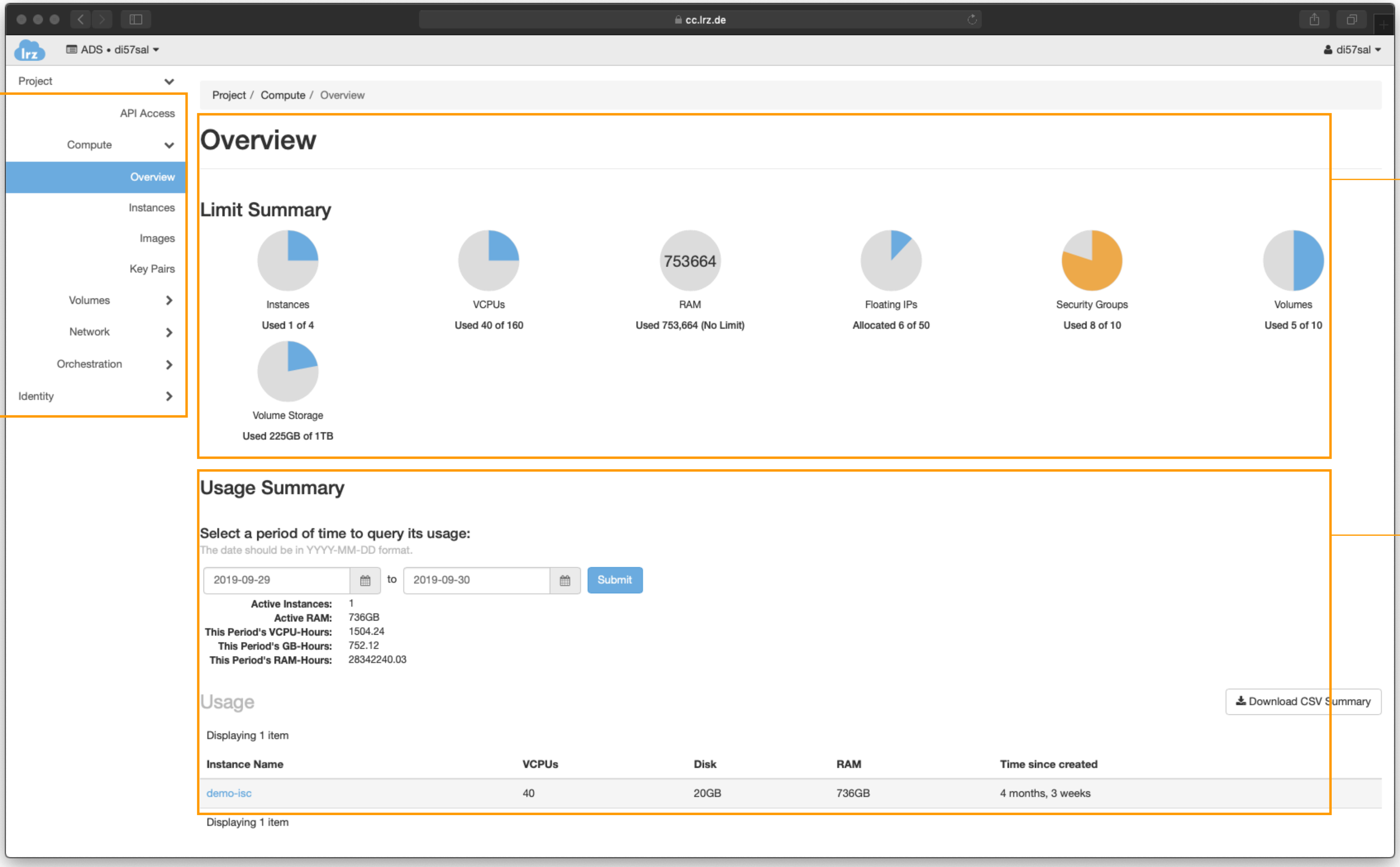
- De-facto operation is no Graphical Interface on the provided images
  - Although it is possible (e.g., https://rv.lrz.de ← only accessible from within MWN)
- Accessing instances via ssh
  - No login based on password by default (public and private keys!)
- OpenStack must be aware of your public key(s) to add it(them) to newly created instances (otherwise you will not be able to login)
  - You can import a public key of a keypair generated using your method of preference
  - You can generate a keypair using OpenStack
    - the private key will be downloaded to your computer
    - the public will be recorded by OpenStack

The Compute Cloud Web Interface



The Compute Cloud Web Interface

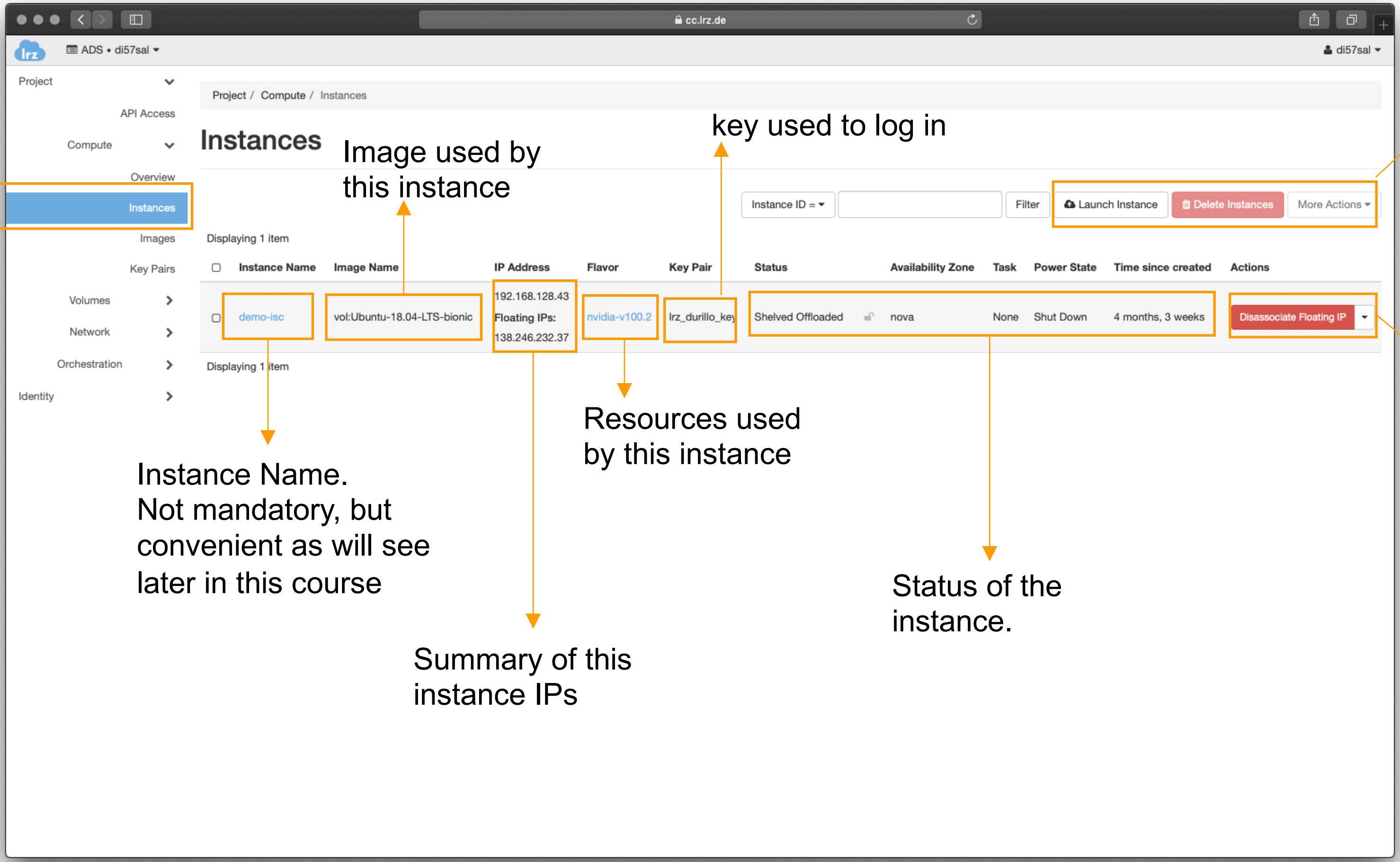
Left panel.
Allows you to operate with the CC.



Resources Limits. Shows you what you are allow and what you have in use.

Summary. Show the instances you have. They might be in different status.

The Compute Cloud Web Interface



Allows a more informative view of our instances

Image used by this instance

key used to log in

Allows creating new instances / Deleting existing ones

Allows operating this instance (e.g., attach a volume, associate new floating IP)

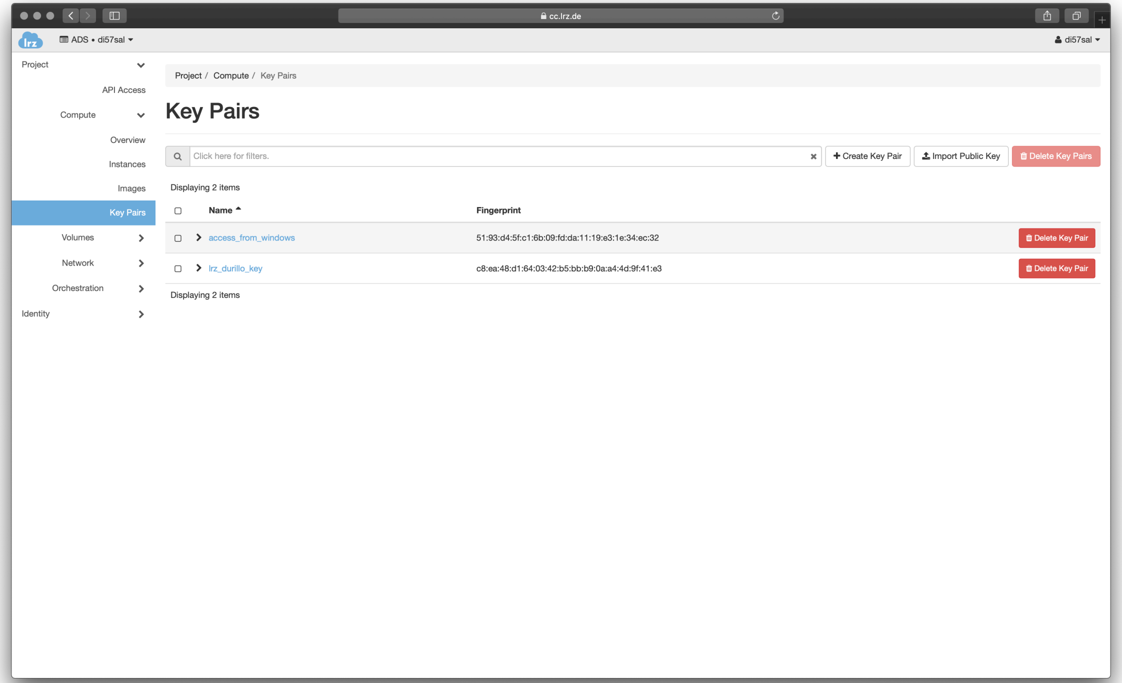
Instance Name. Not mandatory, but convenient as will see later in this course

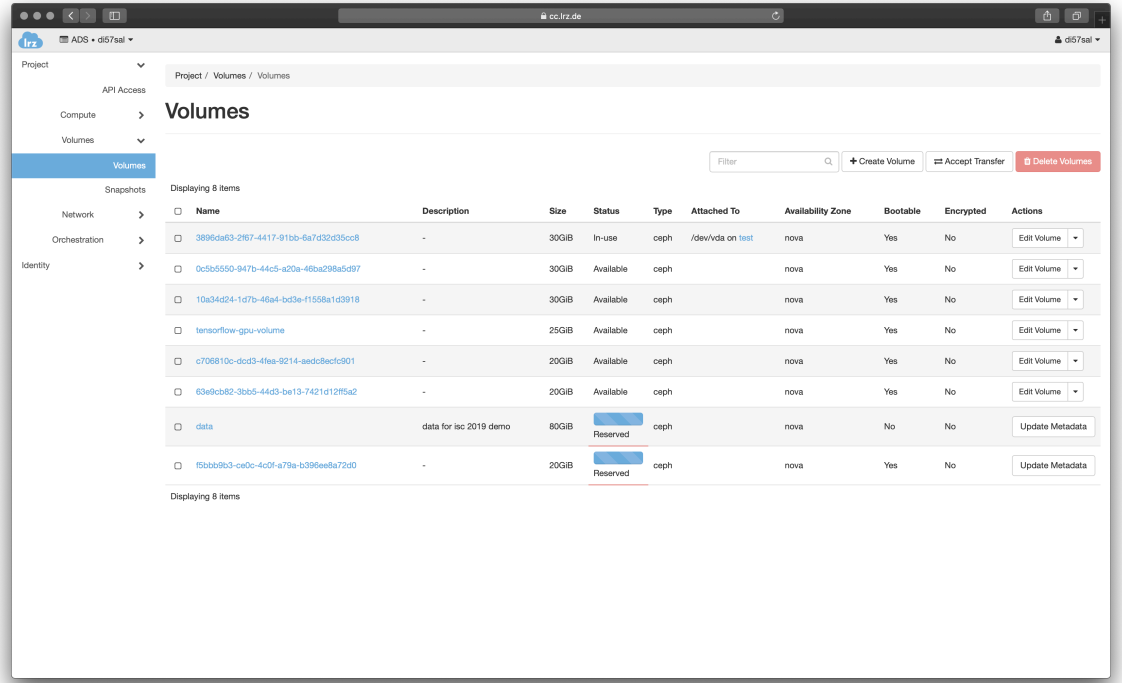
Resources used by this instance

Summary of this instance IPs

Status of the instance.

The Compute Cloud Web Interface





A Guided Example

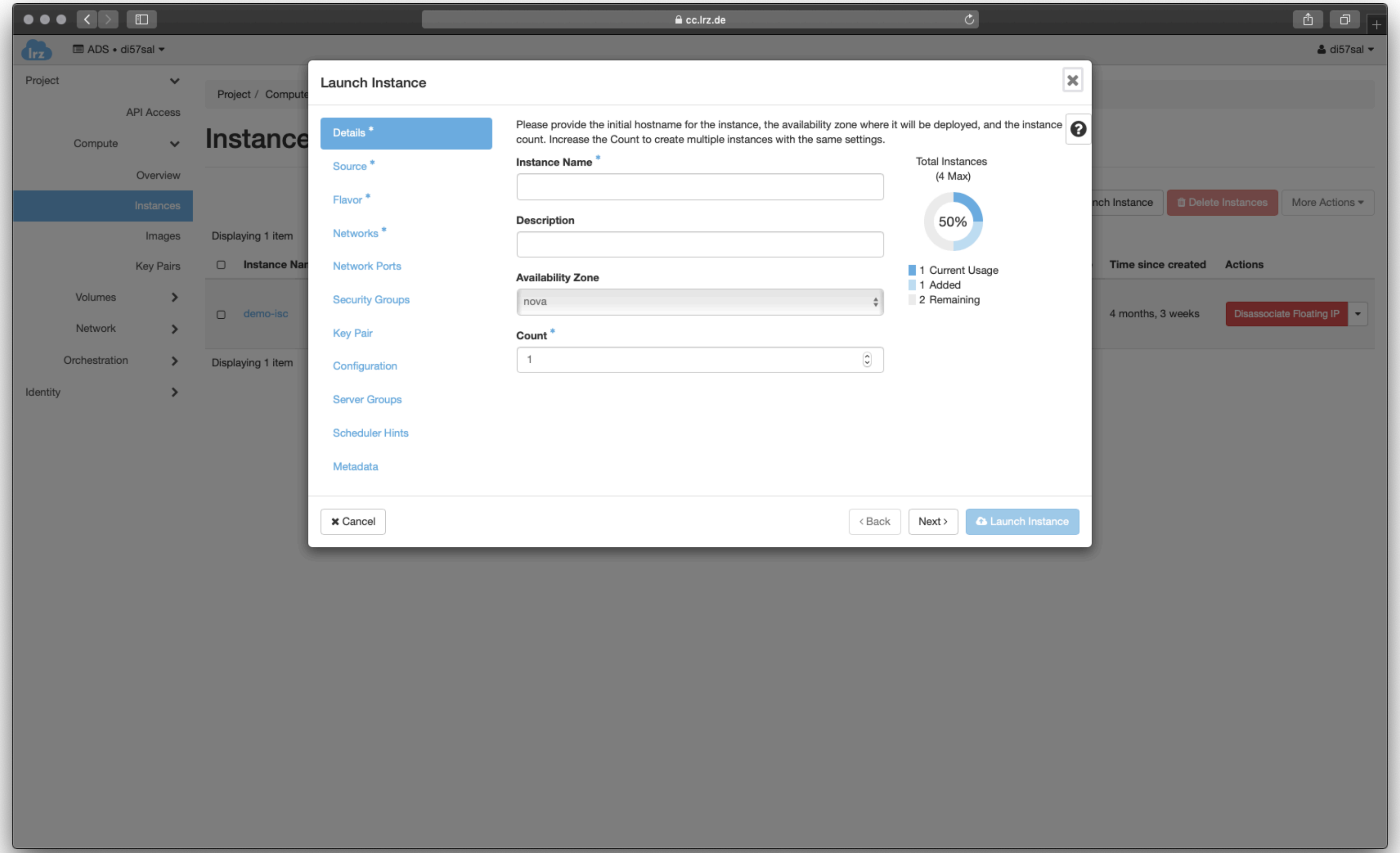
TASK : Create a Ubuntu-based server for running Jupyter Notebooks

- On the Compute Cloud Web Interface
  - Generate a new keypair (follow along the live demo)
- On your computer
  - A file with the extension .pem will be downloaded to your machine (the private key) from previous step
    - **In Linux/UNIX:** change the permission of that file to 600 ($ chmod 600 …)
    - **In Windows with WLS:** copy the downloaded file to inside the WSL (/mnt/c/ allows you accessing C:\ in windows from WSL,)
      - **Once copied, change the permissions as in the Linux/UNIX case**
    - **In Windows with Putty:** import it using PuttyGen
      - check https://stackoverflow.com/questions/3190667/convert-pem-to-ppk-file-format if you need help

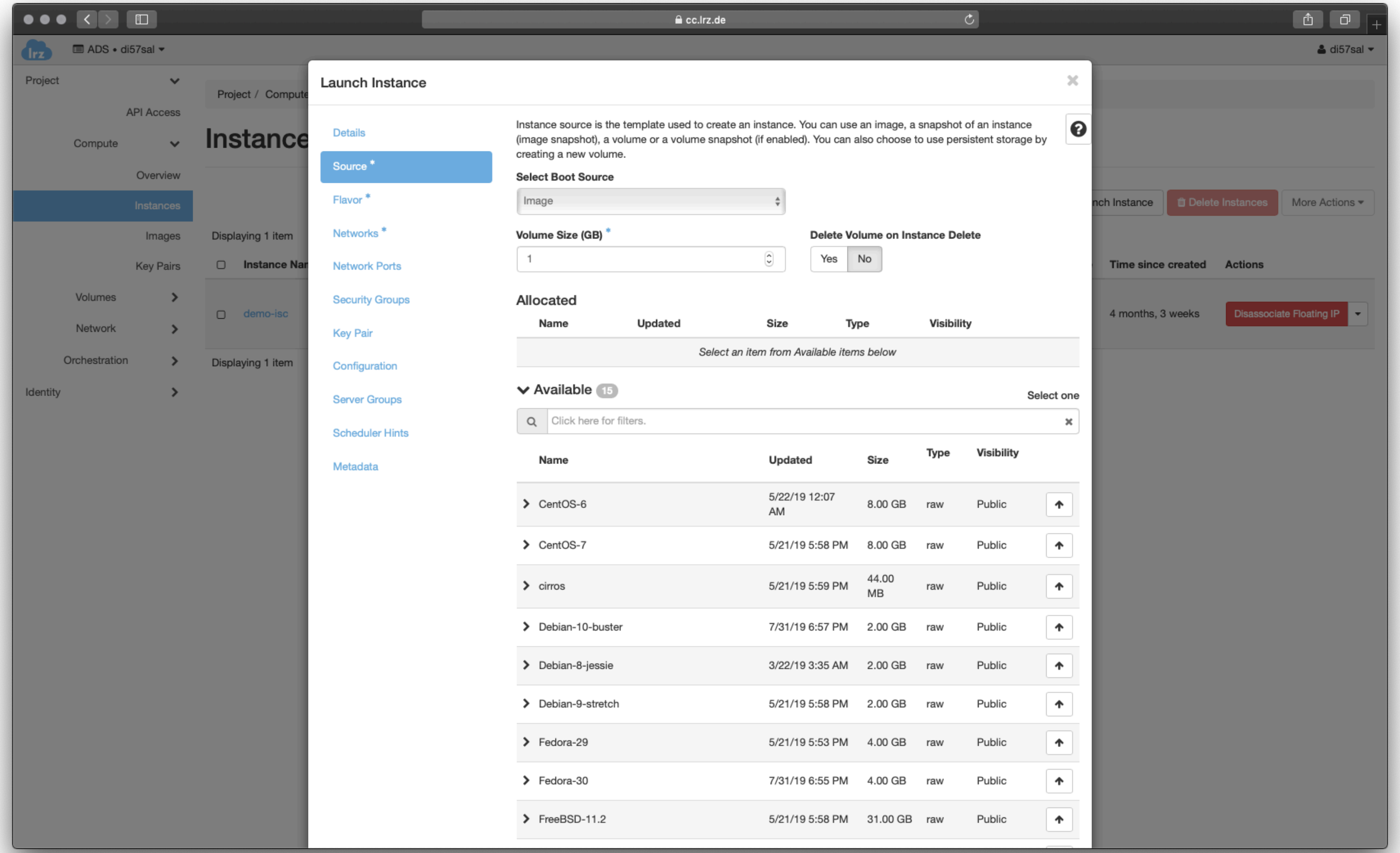
A Guided Example

- On the Compute Cloud Web Interface
  - Create an instance (next steps are documented with screen shots in successive slides)
    - Choose Ubuntu as image
    - CPU only flavor (preferably a small one)
    - Should be accessible from Internet
      - Place the instance on the private network called internet
      - Once the instance is created assign it a floating IP from the Internet pool

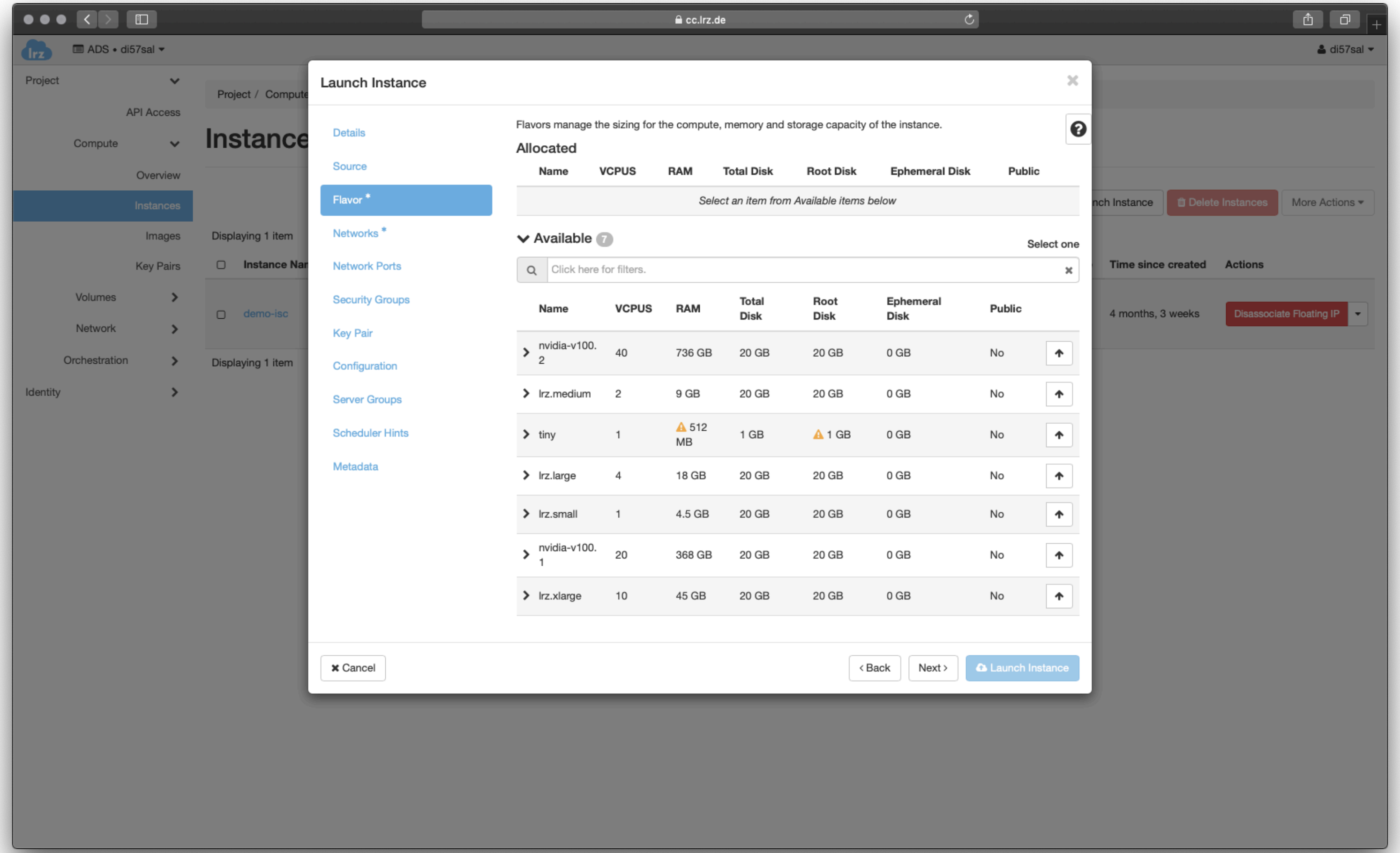
Creating an Instance using the Web Interface



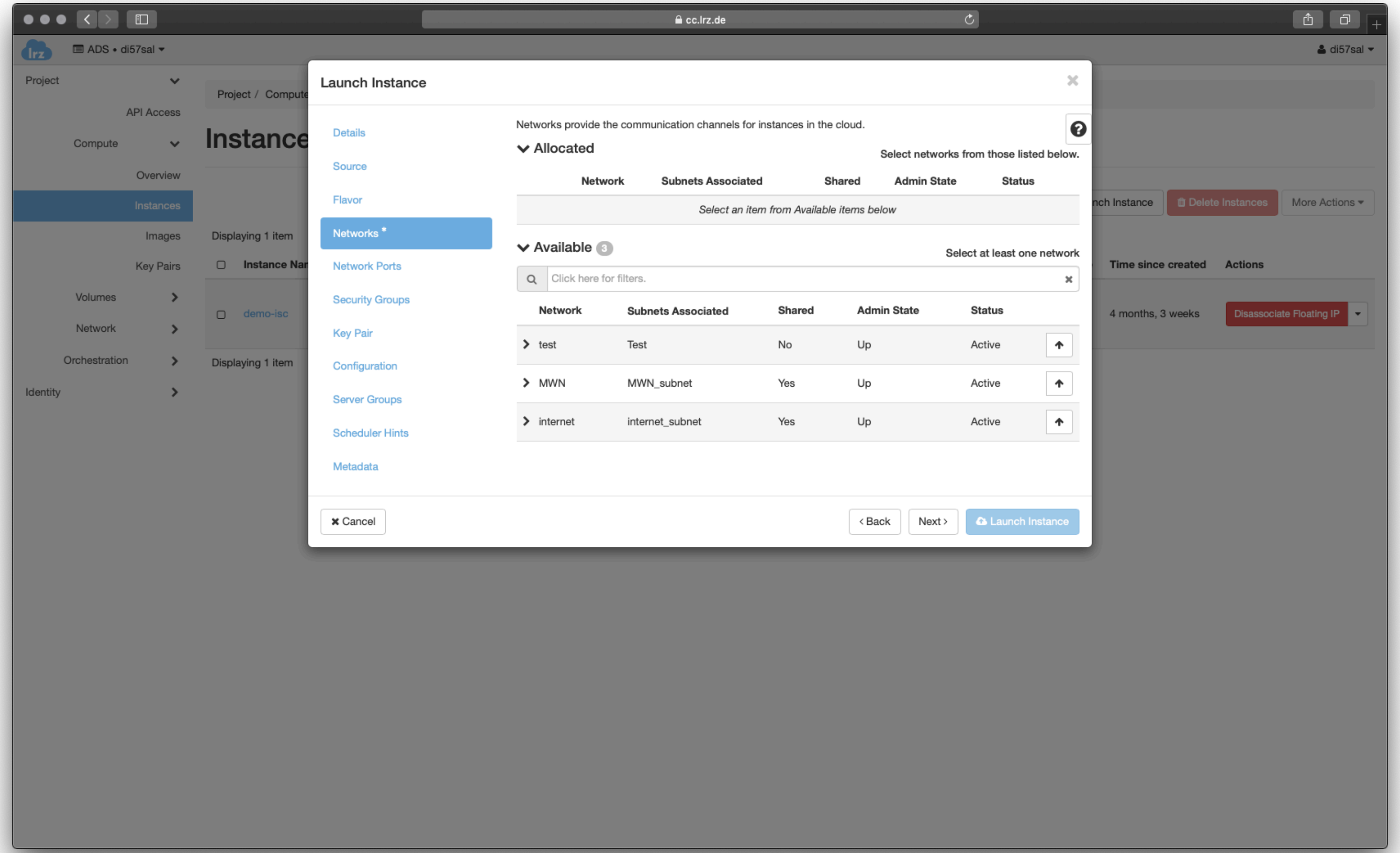
Creating an Instance using the Web Interface



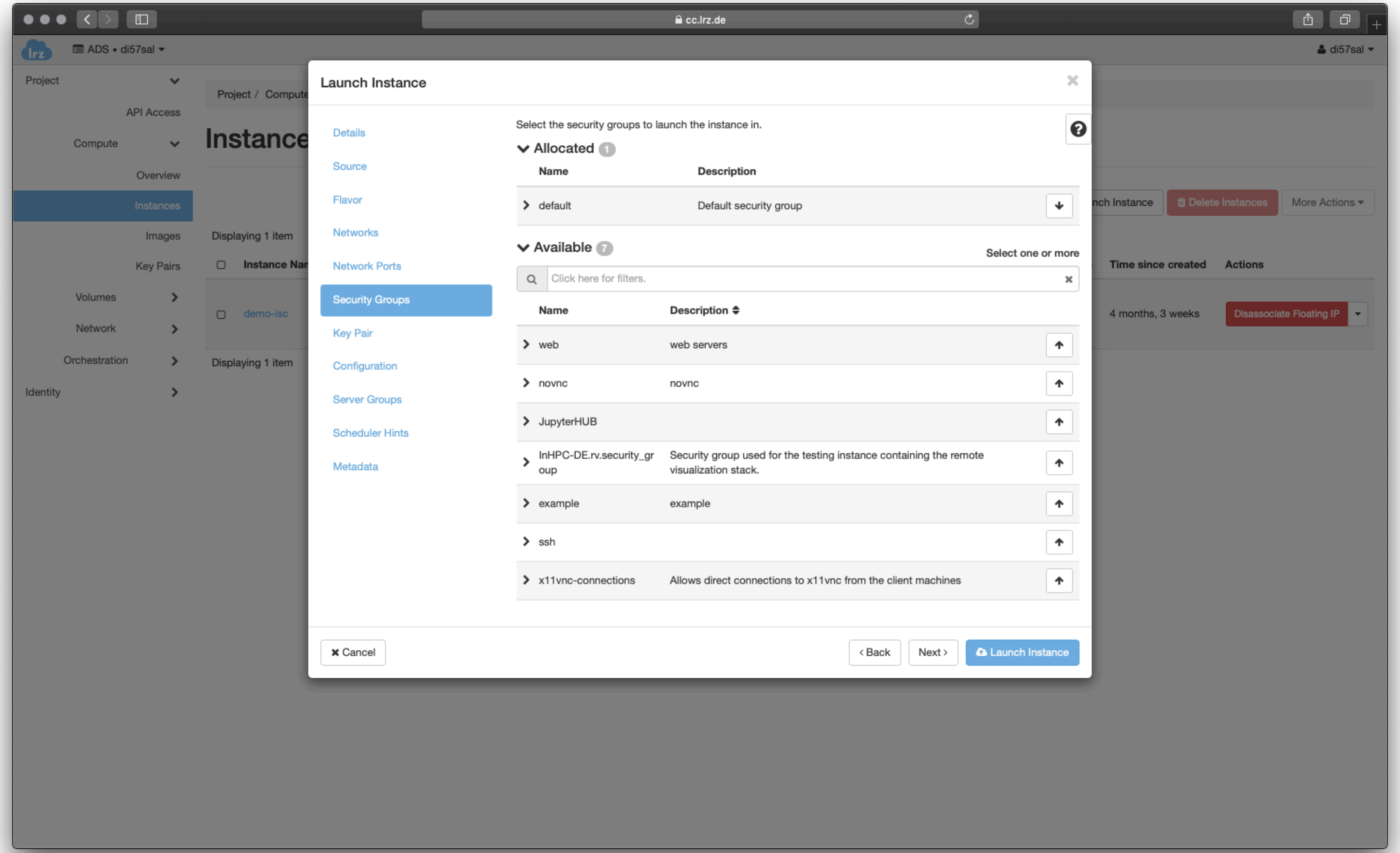
Creating an Instance using the Web Interface



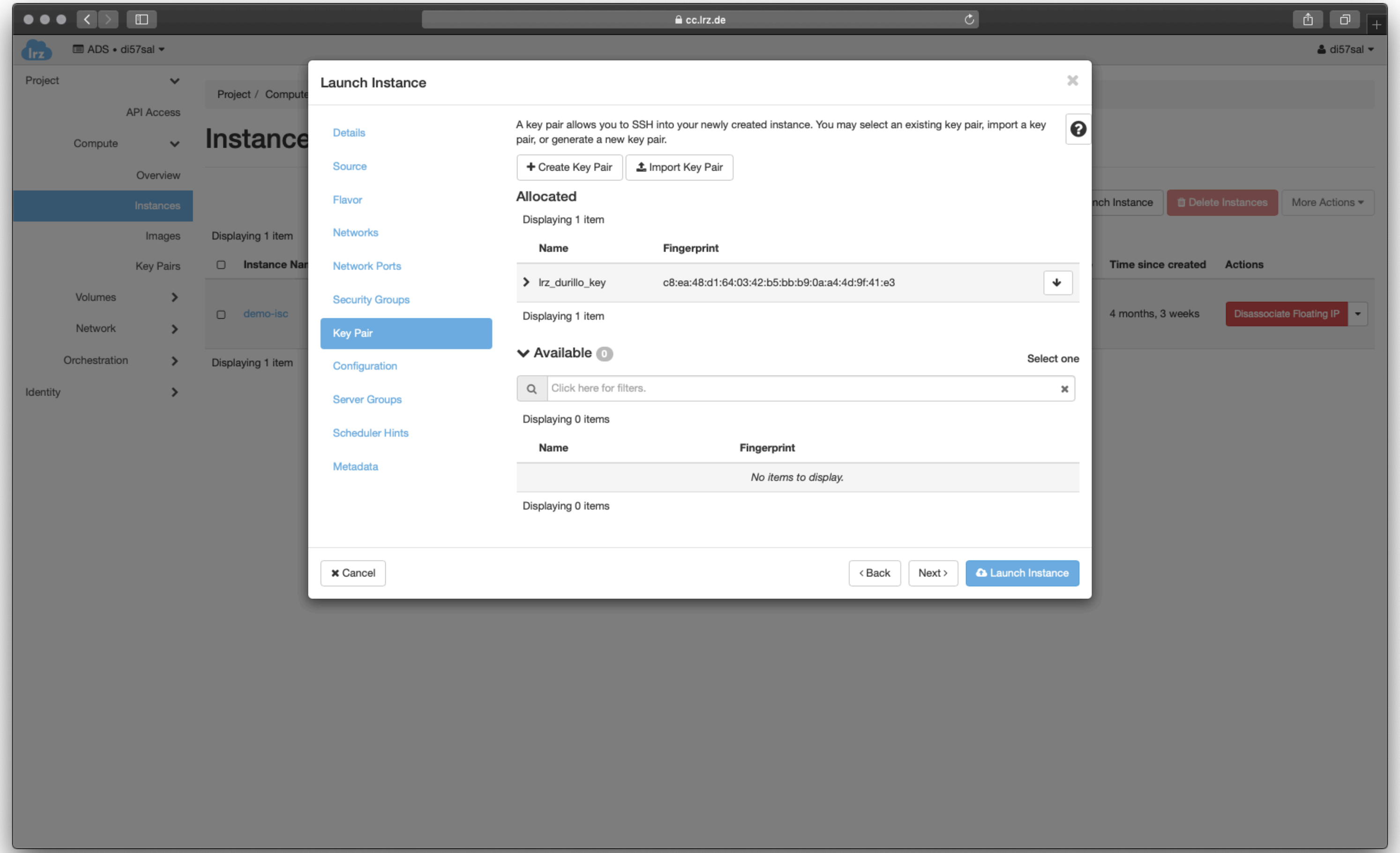
Creating an Instance using the Web Interface



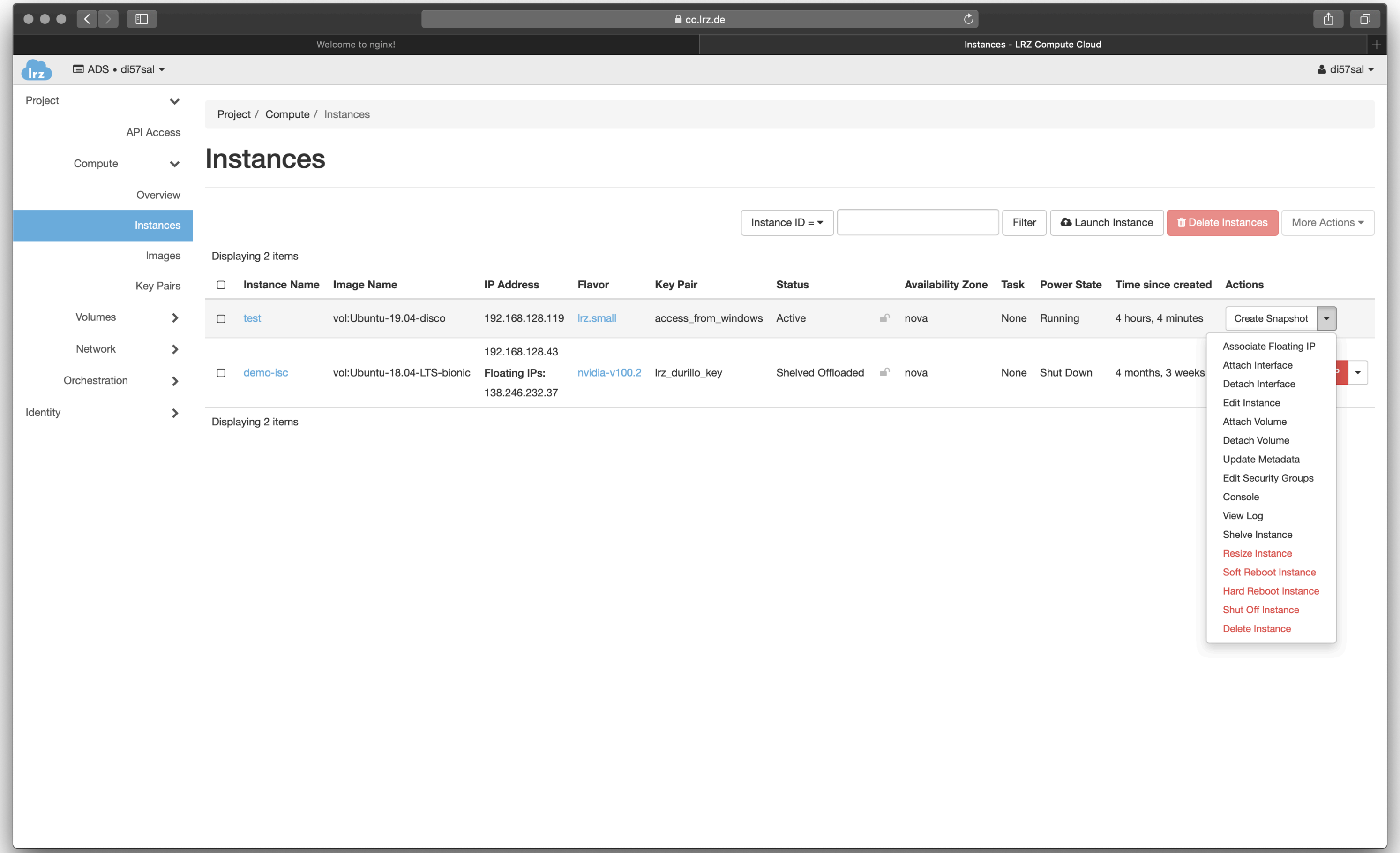
Creating an Instance using the Web Interface



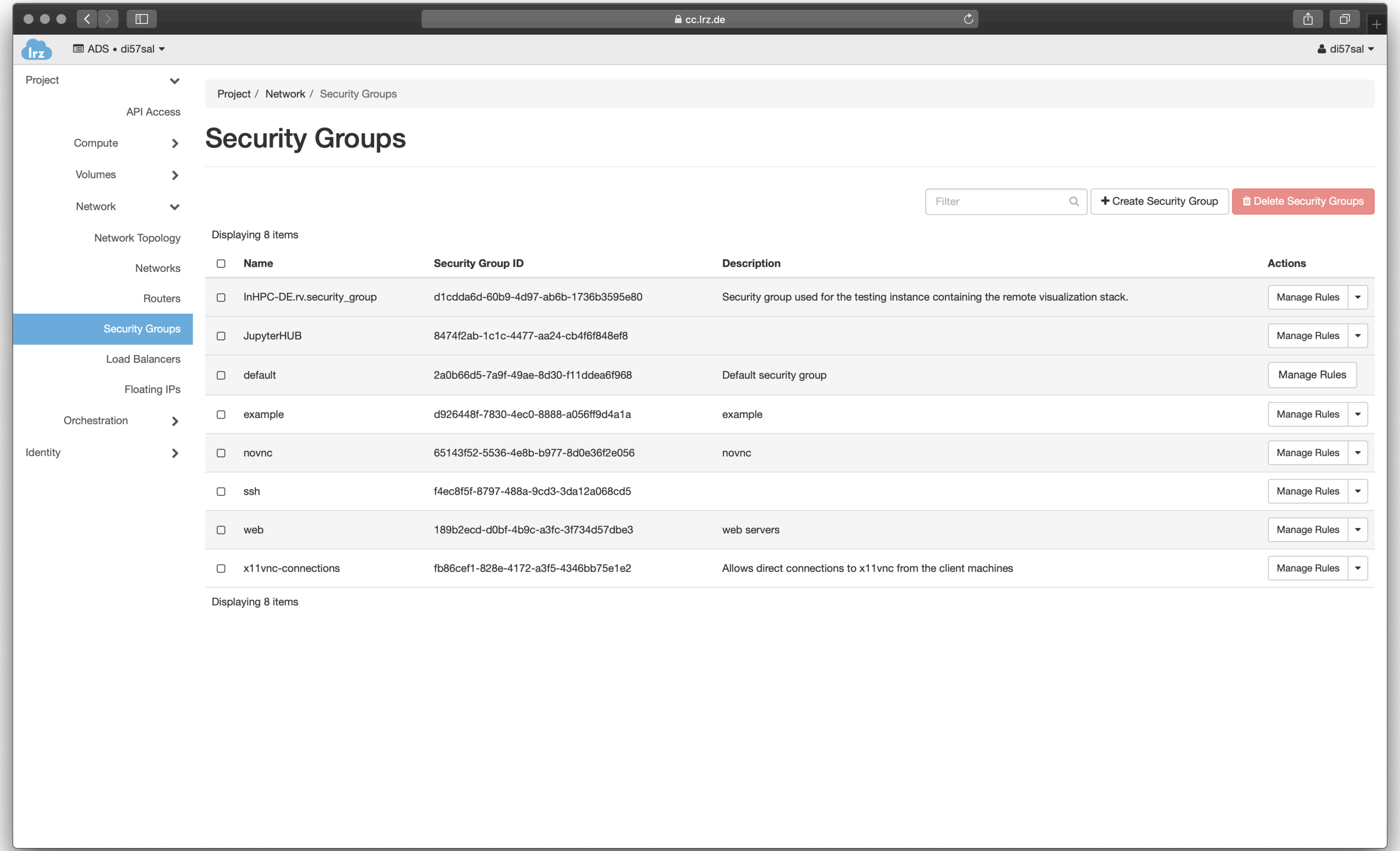
Creating an Instance using the Web Interface



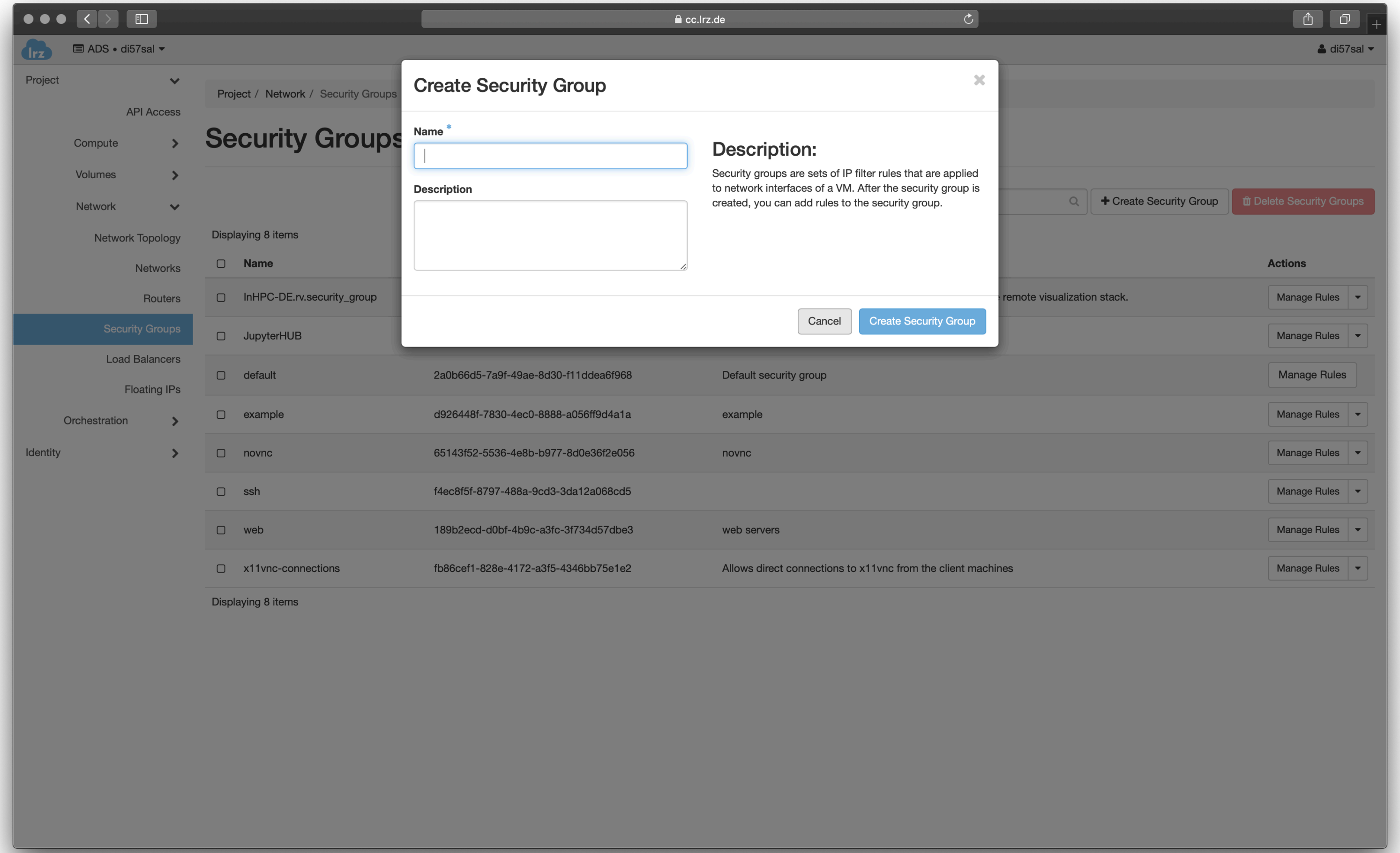
Creating an Instance using the Web Interface



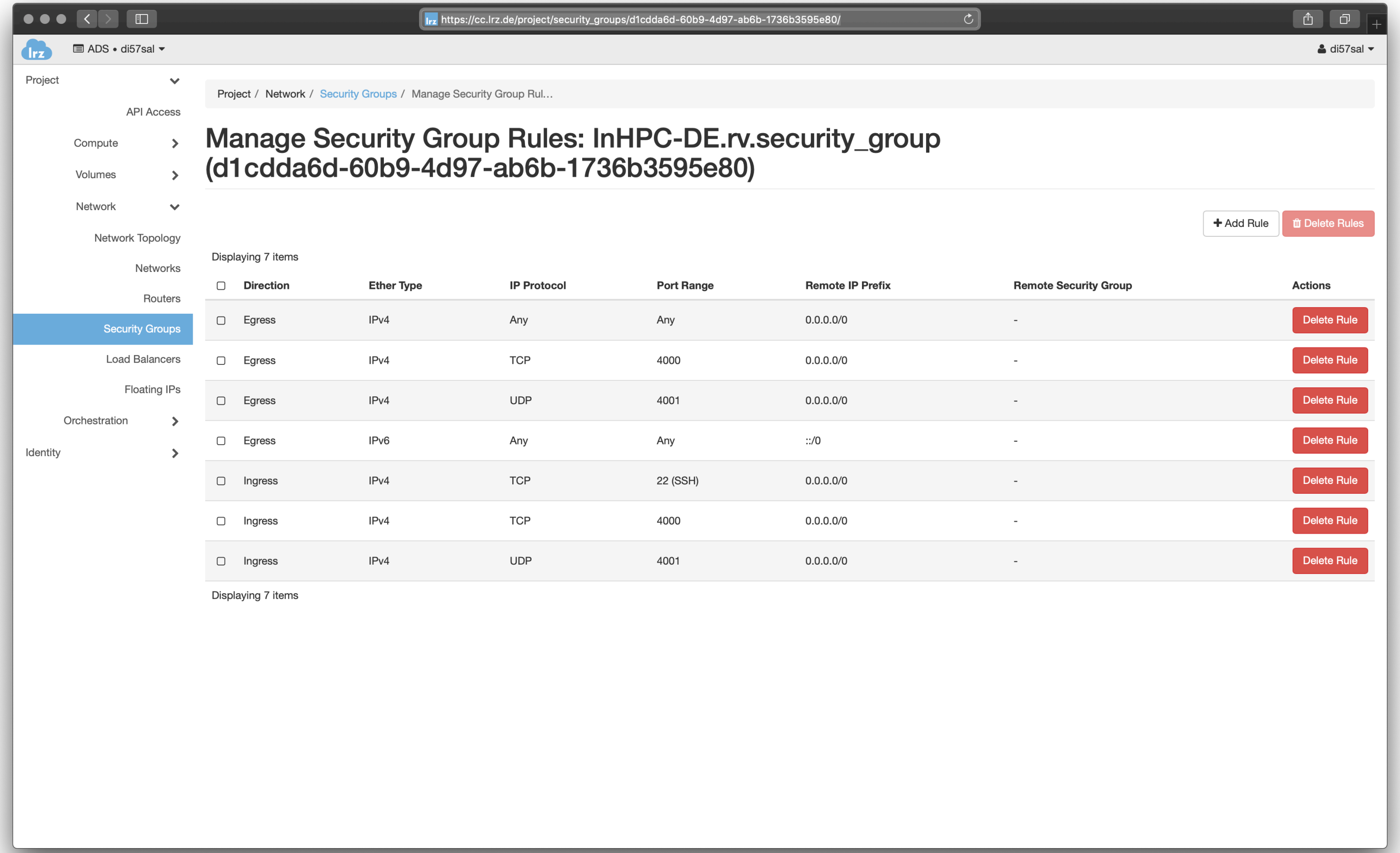
Creating an Instance using the Web Interface



Creating an Instance using the Web Interface



Creating an Instance using the Web Interface



A Guided Example

Subtask: Access the created instance via SSH

- On the Computer Cloud Web Interface
  - **Create a security group that allow ingress connections to port 22!**
  - Add this security group to the instance
- On your computer
  - Open a terminal application

```
ssh –i <path_to_the_pem_file> ubuntu@<floating-ip>
```

After this step, the rest of slides assume everyone is connected via ssh to the created instance

A Guided Example

- Install pip

```
$ sudo apt update
$ sudo apt install python3-pip
```

- Install jupyterlab using pip

```
$ pip install jupyterlab
```

- Run jupyter-lab to listen on the private IP of the machine

```
$ jupyter-lab --ip=192.168.3.140
```

You can check the IP using ifconfig (you need to install net-tools)

A Guided Example

- Connect to she showed URL with a browser. Why is not working?

```
$ jupyter-lab --ip=192.168.3.140
 Or copy and paste one of these URLs:

http://192.168.3.140:8888/lab?token=d61e9775638219bcdc0260b1cd58
4767467eed3f7d440ff9
     or
http://127.0.0.1:8888/lab?token=d61e9775638219bcdc0260b1cd584767
467eed3f7d440ff9
```

- On the Compute Cloud Web Interface
  - **Create a security group that allow ingress connections to port 8888!**
  - Add this security group to the instance

A Guided Example

- Connect to she showed URL with a browser. Why is still not working?

```
$ jupyter-lab --ip=192.168.3.140
 Or copy and paste one of these URLs:

http://192.168.3.140:8888/lab?token=d61e9775638219bcdc0260b1cd58
4767467eed3f7d440ff9
     or
http://127.0.0.1:8888/lab?token=d61e9775638219bcdc0260b1cd584767
467eed3f7d440ff9
```

- The showed URL is relative to the Compute Cloud instance private network.
  - Substitute in that URL the IP with the floating IP of the instance and try accessing the instance now

A Guided Example

- Not done: A few libraries are missing for executing our Jupyter Notebook Example

```
$pip3 install torch
$pip3 install datasets
$pip3 install matplotlib
$pip3 install transformers
$pip3 isntall sklearn
```

Summary

- Overview of the LRZ Resources for ML/DL Workloads
- Focus session on the LRZ AI System, a scenario intended for ML/DL training
- Focus session on the LRZ Compute Cloud, a scenario intended for ML/DL development and inference