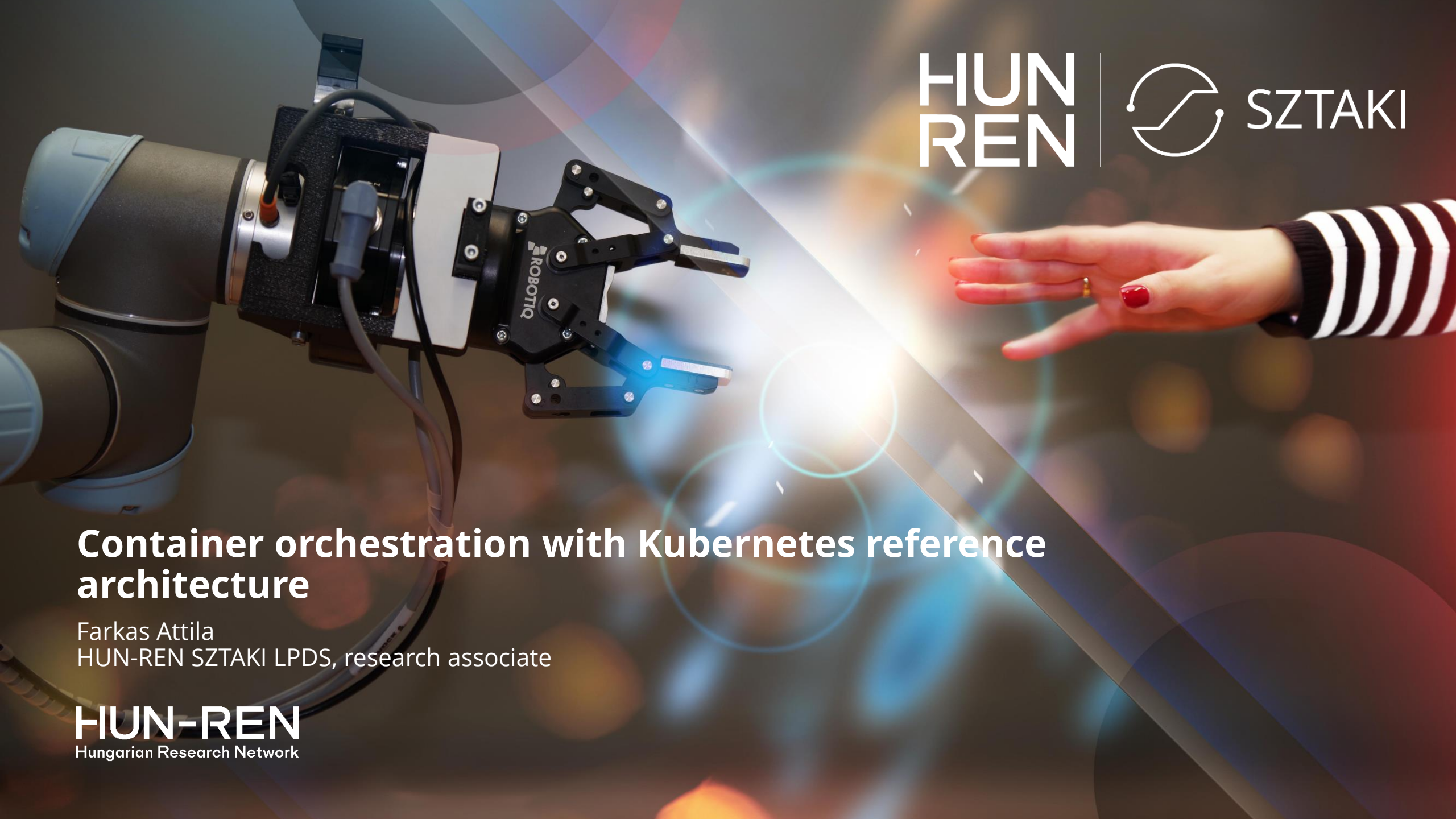


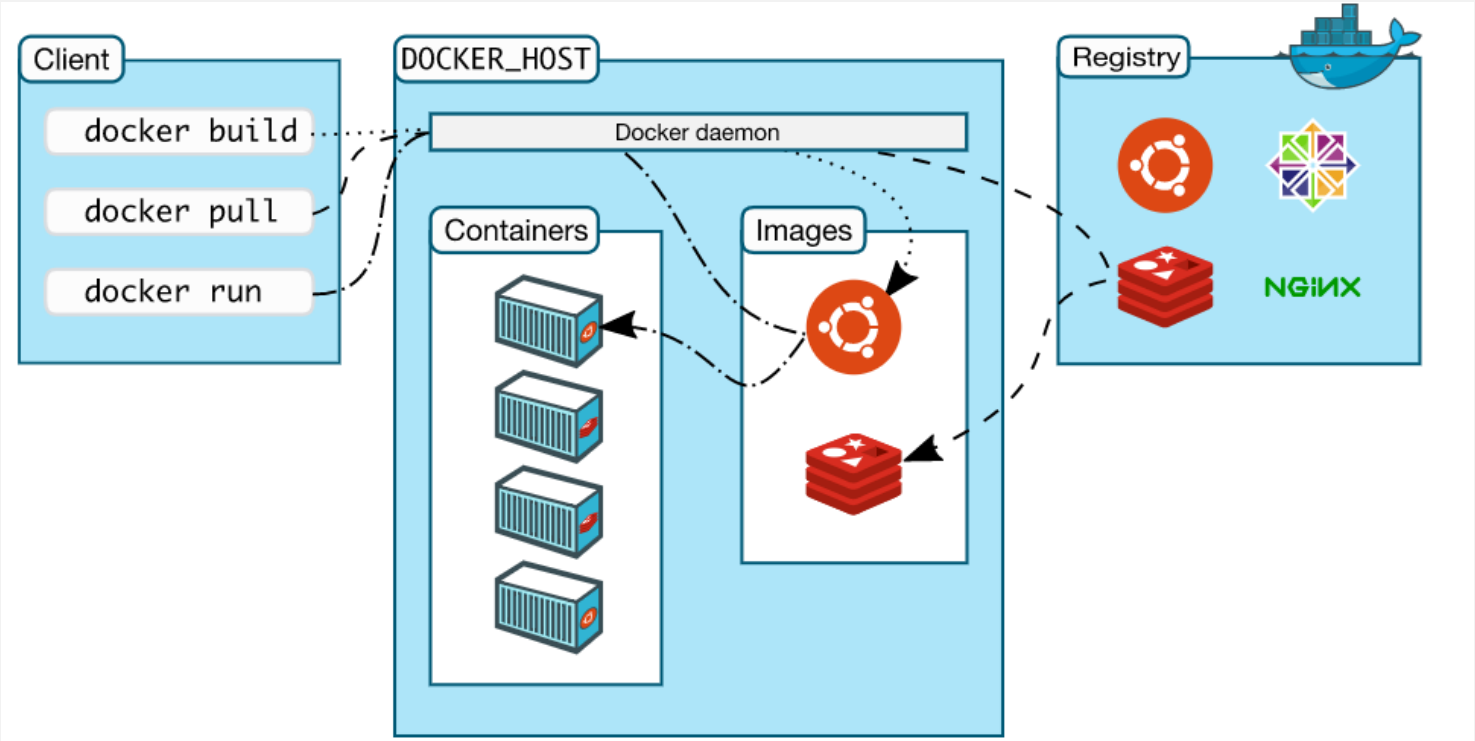
Container orchestration with Kubernetes reference architecture

Farkas Attila
HUN-REN SZTAKI LPDS, research associate

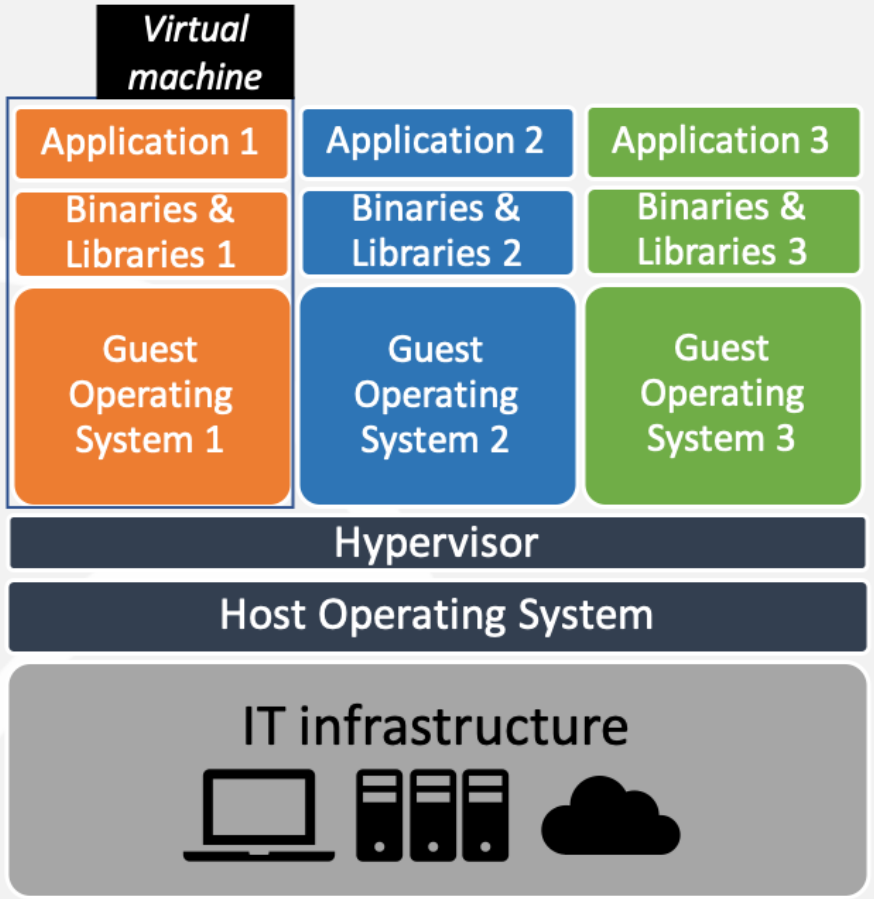
HUN-REN
Hungarian Research Network

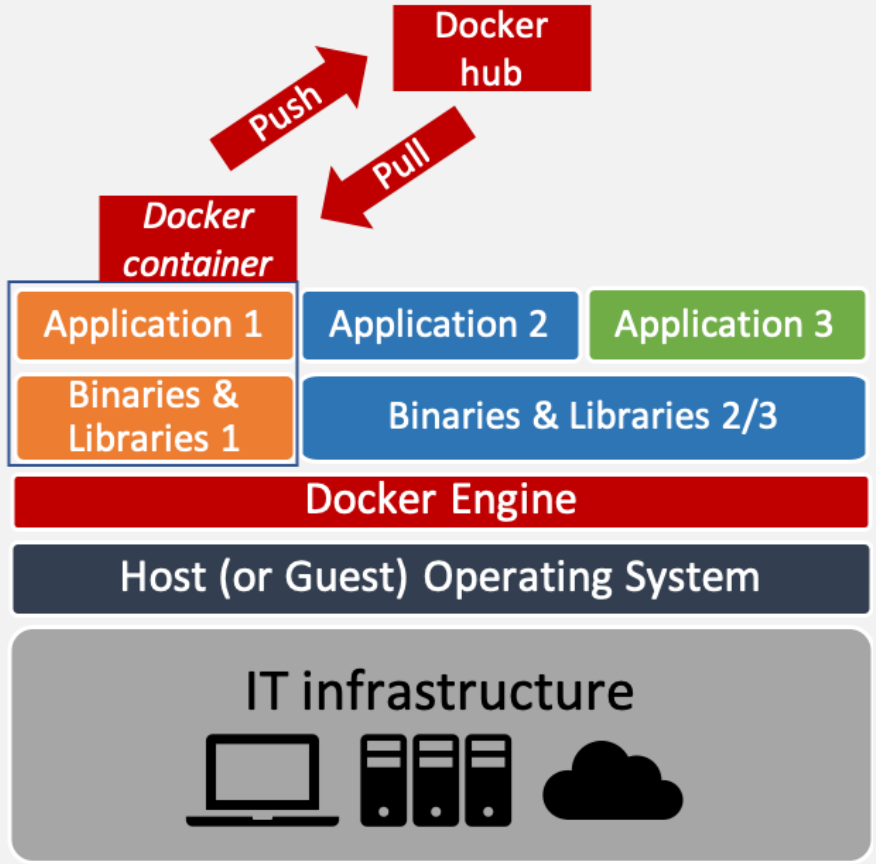
Docker

- Open-source container platform
- Instead of virtualization
- Separated applications
- Based on Linux container technology
- Layered containers
- Centralized image repository
- Portability



Virtual machine vs. Container

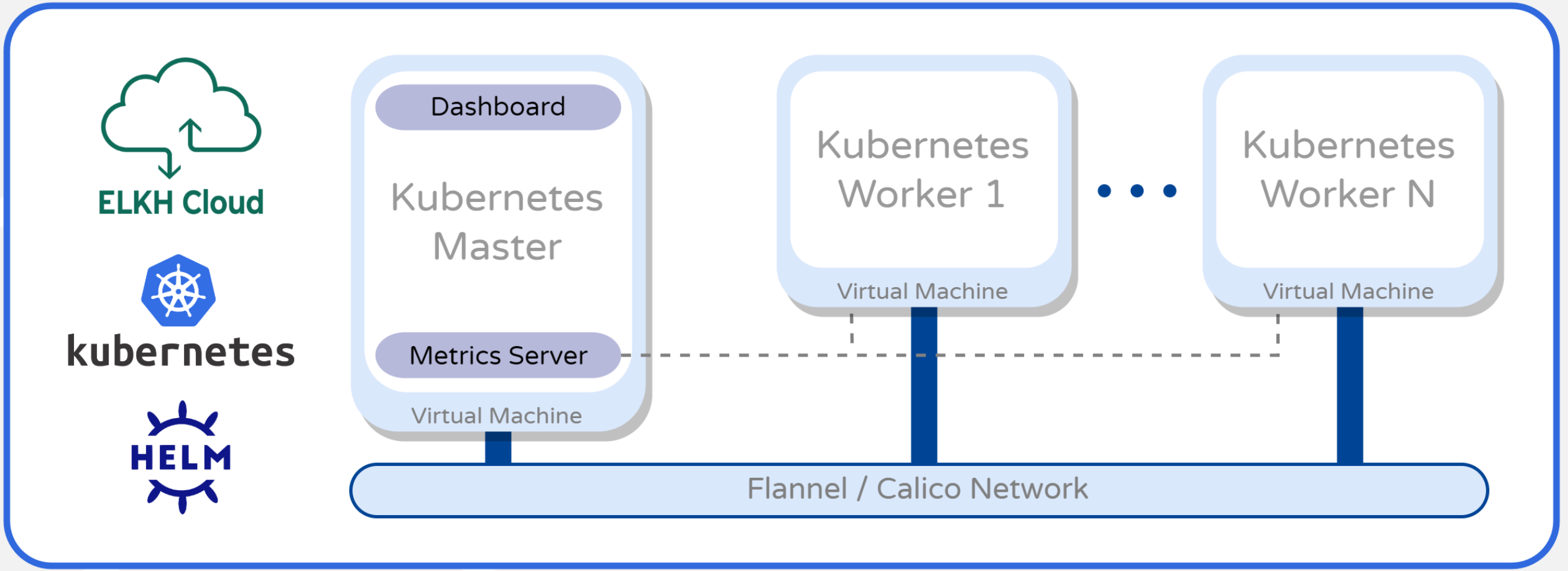




Kubernetes

- Also known as K8s
- Open-source container orchestrator
- Automated rollouts and rollbacks
- Service discovery and load balancing
- Storage orchestration
- Self-healing
- Batch execution
- Horizontal scaling
- Designed for extensibility

Kubernetes reference architecture



https://science-cloud.hu/en/reference-architectures/kubernetes-cluster

HUN REN | SZTAKI

Usage of reference architectures

Necessary steps from the user:

Step 0 - Preparation (HUN-REN Cloud project, creation of empty VM if necessary)

Step 1 - Terraform and Ansible installation

Step 2 - Download the descriptor files
HUN-REN Cloud webpage

Step 3 - Create firewall rules (optional)
HUN-REN Cloud OpenStack dashboard or descriptor file

Step 4 - Personalizing descriptor files

Step 5 - Terraform initialization

```
$ terraform init
```

Step 6 - Infrastructure deployment

```
$ terraform apply (--auto-approve)
```

Step 7 - Infrastructure usage

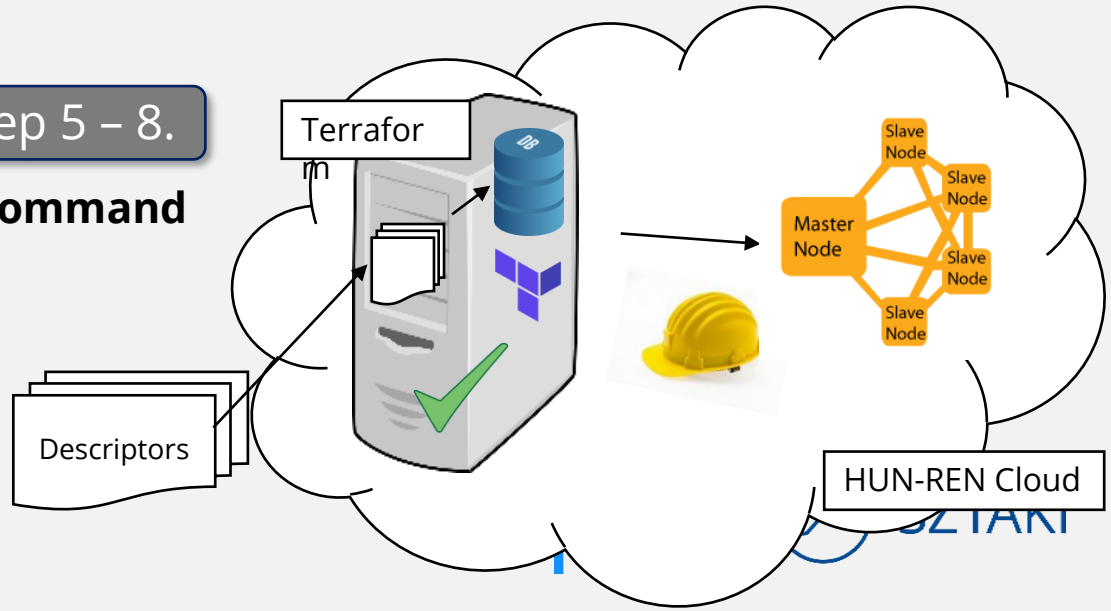
Step 8 - Infrastructure destroy

```
$ terraform destroy (--auto-approve)
```

Step 0 – 1. **Only for the first time.**

Step 2 – 4. **Only 1 time per reference architecture**

Step 5 – 8. **1 command**



Step 1 - Terraform and Ansible installation

- Terraform and Ansible installation

```
sudo apt-get update && sudo apt-get install -y gnupg software-properties-common curl
curl -fsSL https://apt.releases.hashicorp.com/gpg | sudo apt-key add –
sudo apt-add-repository "deb [arch=amd64] https://apt.releases.hashicorp.com $(lsb_release -cs) main„
sudo apt-get update && sudo apt-get install terraform=1.3.6

sudo apt install python3-pip
sudo python3 -m pip install ansible==6.7.0
```

- (Optional) RefArch Toolset Docker image

```
docker run -it -v PATH_TO_WORKDIR:/home/refarch -v PATH_TO_PRIVATE_KEY:/root/.ssh/id_rsa:ro
git.sztaki.hu:5050/science-cloud/reference-architectures/refarch-toolset bash
```

Step 2 - Download the descriptor files

HashiCorp Terraform

- **Provision infrastructure**
  - Virtual machines
  - Network settings
  - Firewall rules
  - Execute tasks
  - Invoke Ansible

ANSIBLE

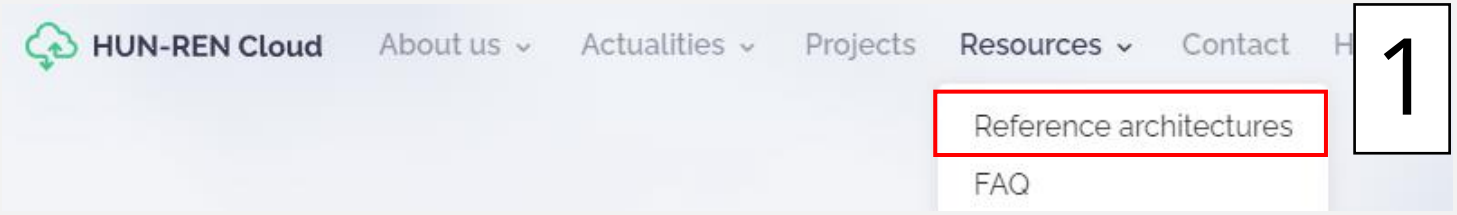
- **Configure nodes**
  - Install packages
  - Start services
  - Run Docker containers

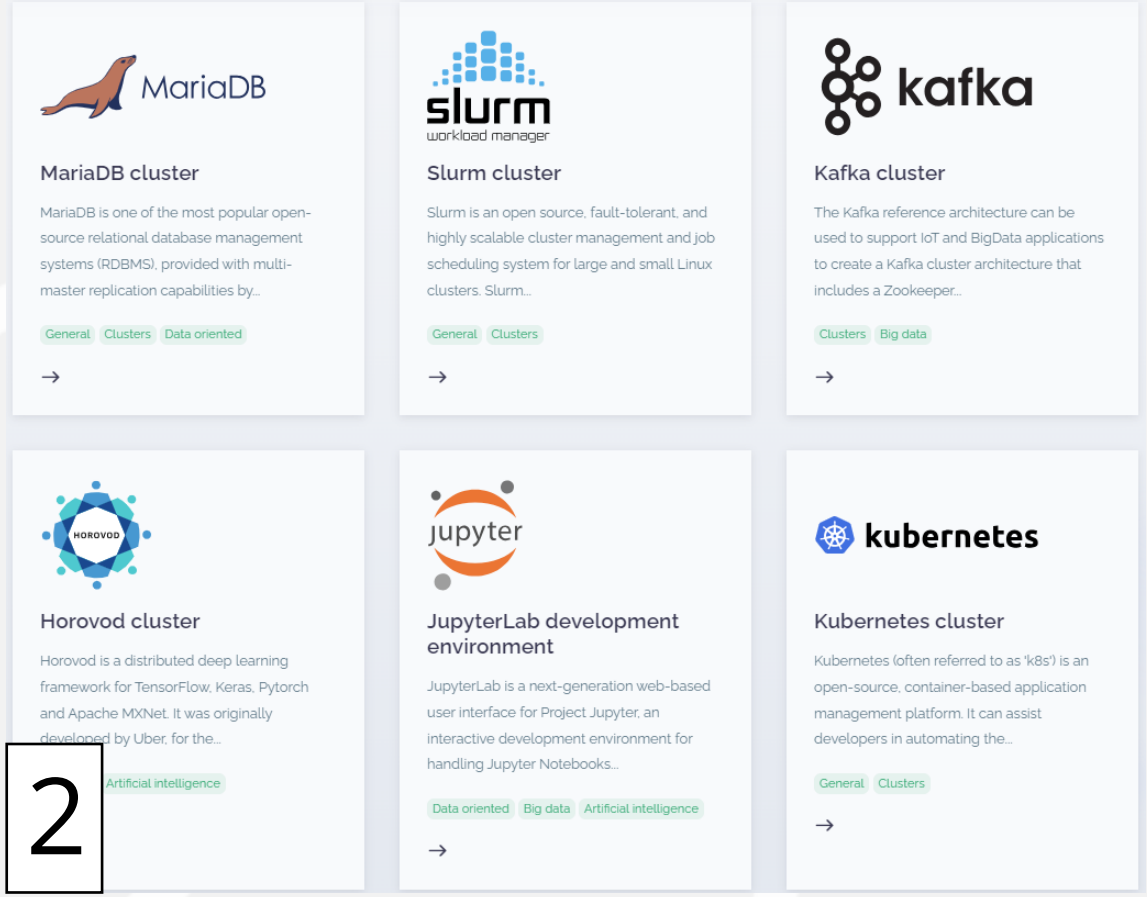
I. Set authentication information
II. Set resource parameters
III. Customize through variables
IV. Deploy

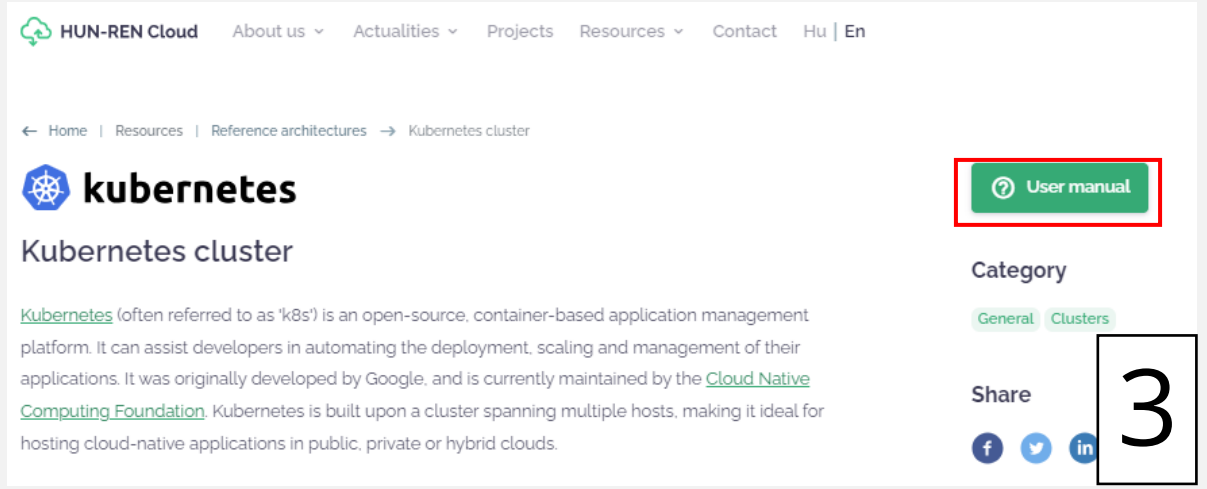
```
horovod_master_node = ({
    name = "horovod-master"
    flavor_name = "SET_YOUR_FLAVOR_NAME"
    image_id = "SET_YOUR_IMAGE_ID"
    key_pair = "SET_YOUR_KEY_PAIR_NAME"
    floating_ip = "SET_YOUR_
    volume_size = 32
})
```

```
user_config = ({
    jupyter_password = "elkhcloud"
    enable_gpu = true
    monitoring = false
    monitoring_password = "elkhcloud"
    nvidia_driver_install = false
    elkh_cloud_dedicated_network = false
})
```

Step 2 - Download the descriptor files







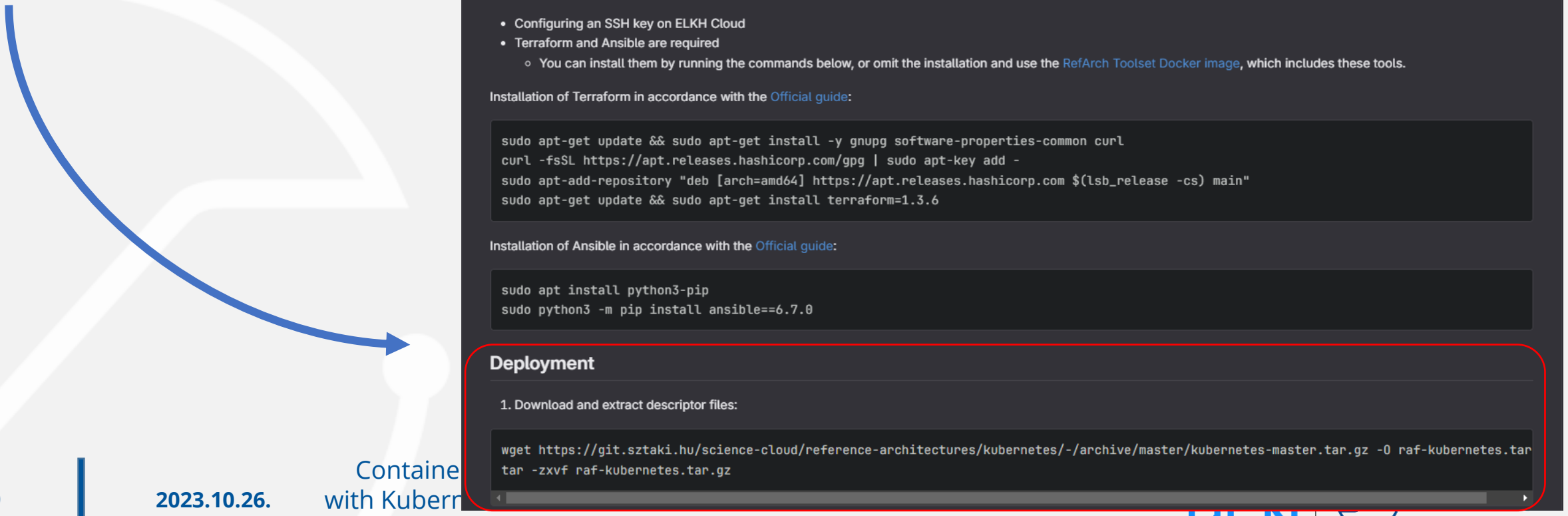
https://science-cloud.hu/en/reference-architectures

HUN REN | SZTAKI

Step 2 - Download the descriptor files

Kubernetes cluster user manual:

https://git.sztaki.hu/science-cloud/reference-architectures/kubernetes



Prerequisites

- Configuring an SSH key on ELKH Cloud
- Terraform and Ansible are required
  - You can install them by running the commands below, or omit the installation and use the RefArch Toolset Docker image, which includes these tools.

Installation of Terraform in accordance with the Official guide:

```
sudo apt-get update && sudo apt-get install -y gnupg software-properties-common curl
curl -fsSL https://apt.releases.hashicorp.com/gpg | sudo apt-key add -
sudo apt-add-repository "deb [arch=amd64] https://apt.releases.hashicorp.com $(lsb_release -cs) main"
sudo apt-get update && sudo apt-get install terraform=1.3.6
```

Installation of Ansible in accordance with the Official guide:

```
sudo apt install python3-pip
sudo python3 -m pip install ansible==6.7.0
```

Deployment

1. Download and extract descriptor files:

```
wget https://git.sztaki.hu/science-cloud/reference-architectures/kubernetes/-/archive/master/kubernetes-master.tar.gz -O raf-kubernetes.tar
tar -zxvf raf-kubernetes.tar.gz
```

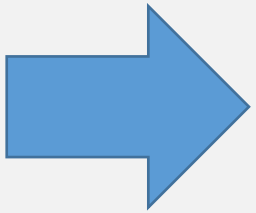
Step 4 – Personalizing descriptor files

- Auth definition file (**terraform_openstack/auth_data.auto.tfvars**):

```
auth_data = ({
    auth_url                      = "SET_YOUR_AUTH_URL"
    application_credential_id     = "SET_YOUR_APPLICATION_CREDENTIAL_ID"
    application_credential_secret = "SET_YOUR_APPLICATION_CREDENTIAL_SECRET"
})
```

- Node definition file (**terraform_openstack/resources.auto.tfvars**):

```
kubernetes_master_node = ({
  name          = „kubernetes-master"
  flavor_name   = "SET_YOUR_FLAVOR_NAME"
  floating_ip   = "SET_YOUR_FLOATING_IP"
  image_id      = "SET_YOUR_IMAGE_ID"
  key_pair      = "SET_YOUR_KEY_PAIR_NAME"
  volume_size   = 32
})

kubernetes_worker_node = ({
  name          = „kubernetes-worker"
  count         = 1
  flavor_name   = "SET_YOUR_FLAVOR_NAME"
  image_id      = "SET_YOUR_IMAGE_ID"
  key_pair      = "SET_YOUR_KEY_PAIR_NAME"
  volume_size   = 32
})
```



```
kubernetes_master_node = ({
  name          = „ kubernetes-master"
  flavor_name   = “m2.large"
  floating_ip   = "193.225.250.4"
  image_id      = "11bf8b51-63d3-4277-b3dc-387f4c4bfd0a"
  key_pair      = „farkas"
  volume_size   = 32
})

kubernetes_worker_node = ({
  name          = „ kubernetes-worker"
  count         = 2
  flavor_name   = “m2.large"
  image_id      = "11bf8b51-63d3-4277-b3dc-387f4c4bfd0a"
  key_pair      = „farkas"
  volume_size   = 32
})
```

Step 4 – Personalizing descriptor files

- Set Kubernetes network

```
kubernetes_network = ({
  name  = „SET_YOUR_NETWORK_NAME"
  network_subnet_range  = „SET_YOUR_NETWORK_RANGE"
})
```

- Set Kubernetes software versions

```
software_version = ({
  kubernetes_version  = „1.26"
  containerd_version  = „1.6"
})
```

- Set user configuration

```
user_config = ({
  container_network_interface  = „flannel"
  allow_pods_on_master  = true
  enable_gpu  = false
  install_prometheus_stack  = true
})
```

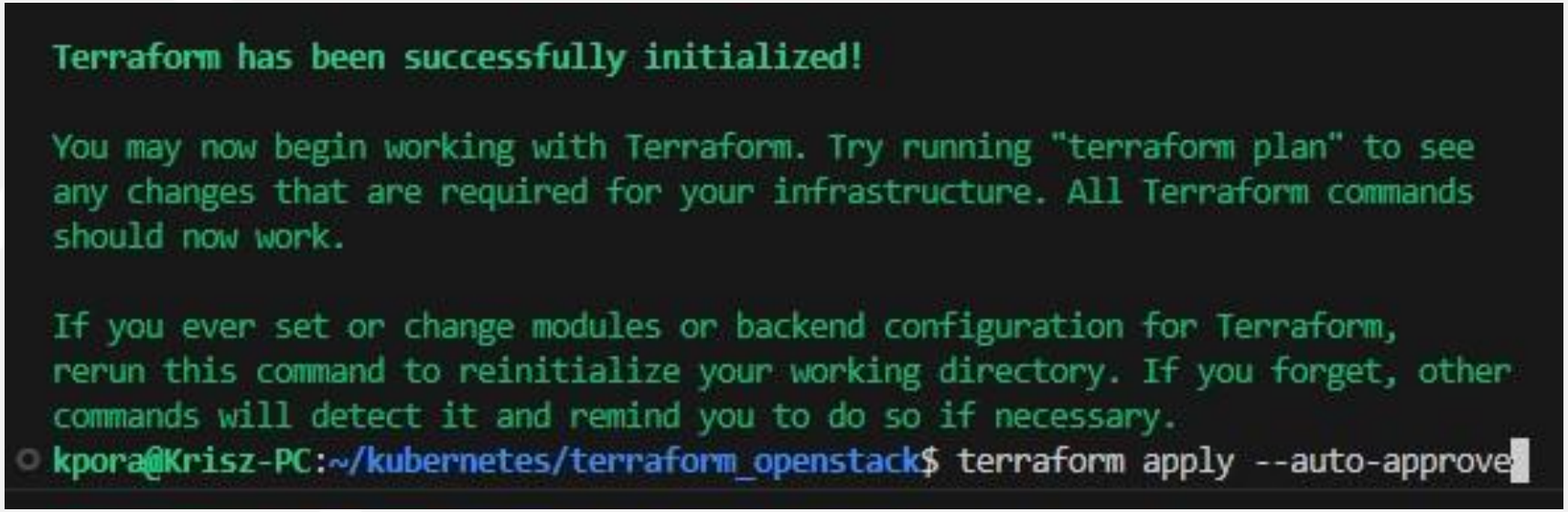
Step 5 - Terraform initialization

Adding the private SSH key to the SSH agent:

```
# eval $(ssh-agent -s) && echo "$(cat PATH_TO_YOUR_KEY)" | tr -d '\r' | ssh-add -
```

Terraform initialization:

```
# terraform init
```

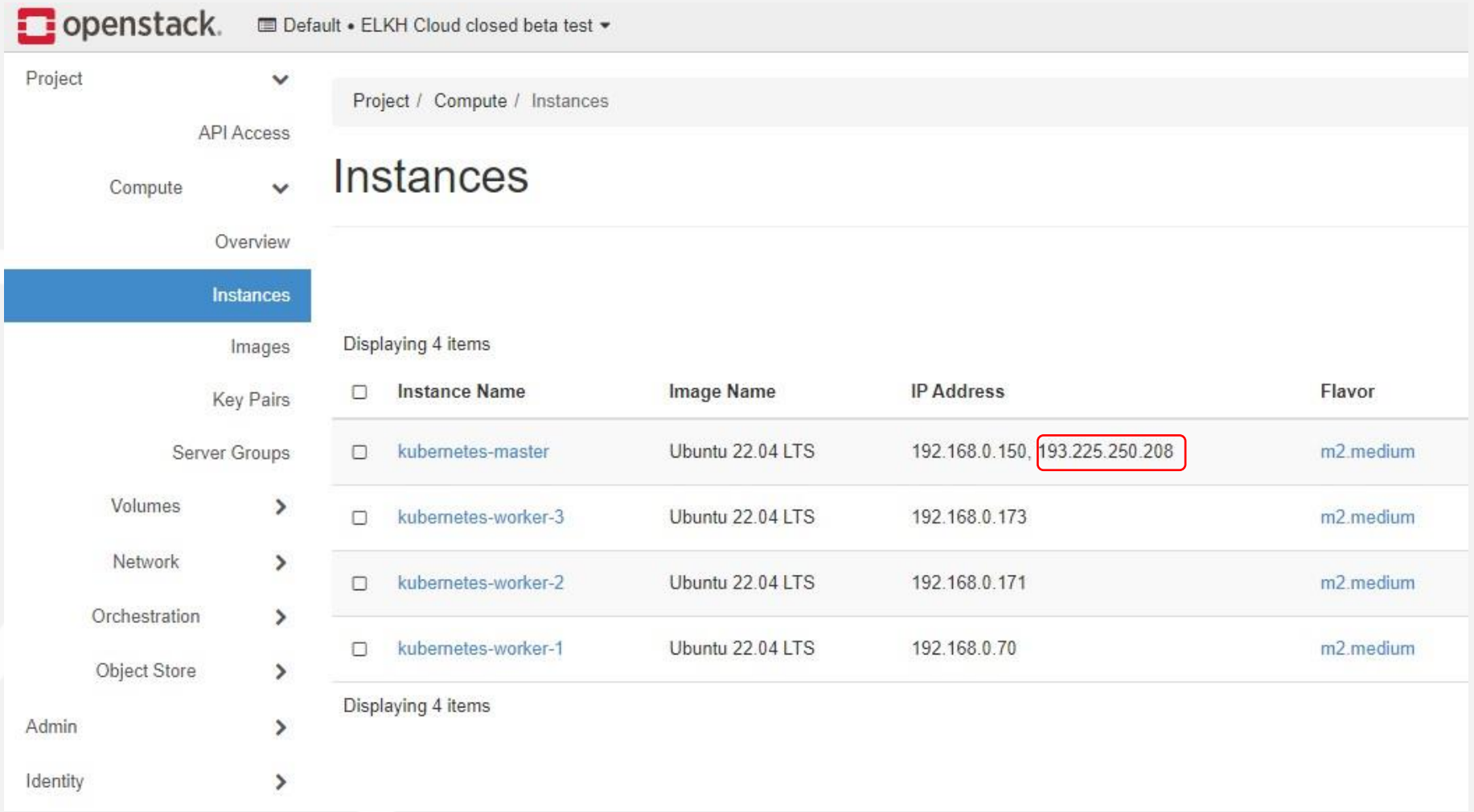


Step 6 - Infrastructure deployment

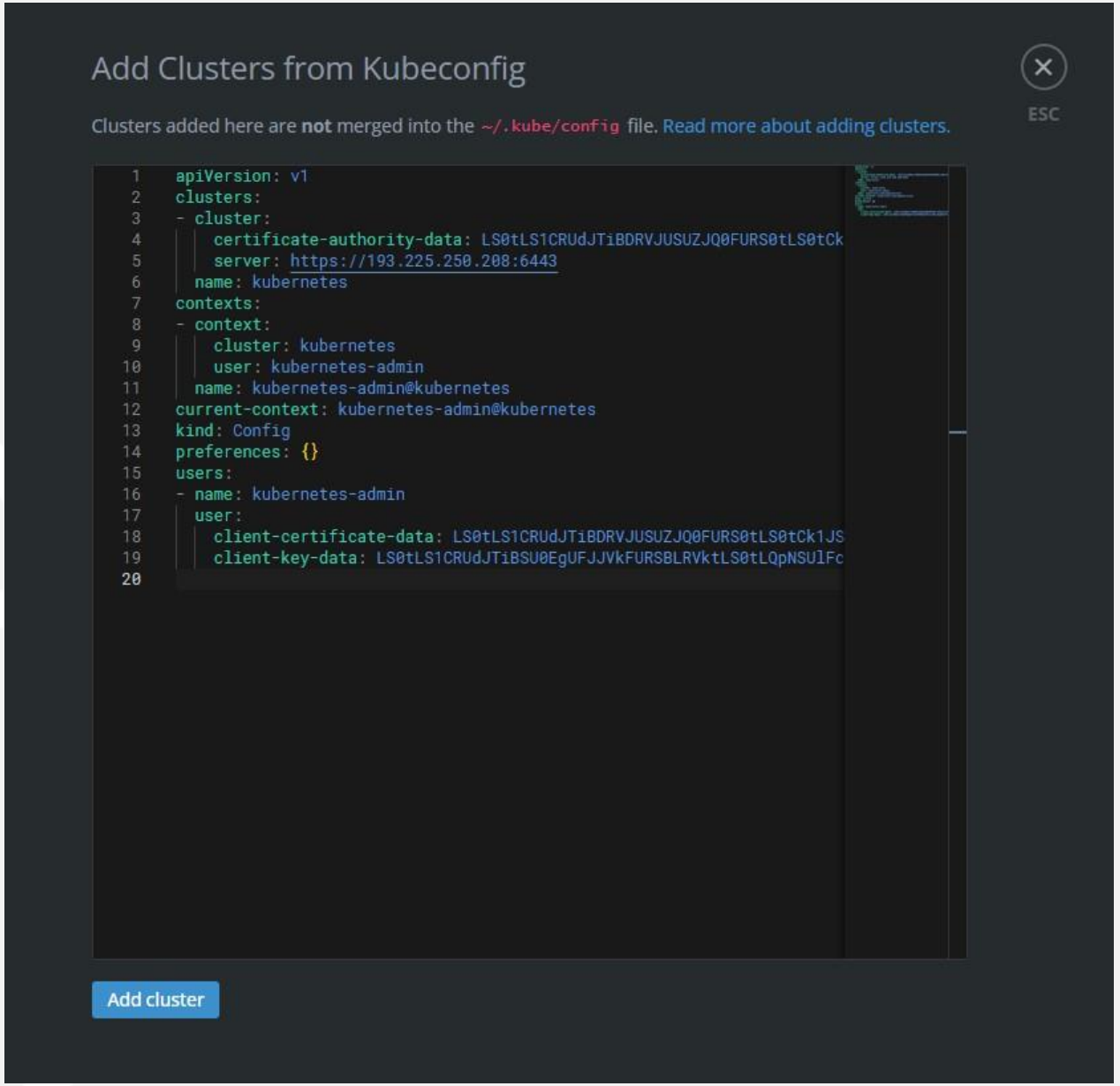
```
# terraform apply --auto-approve
```

```
null_resource.kubernetes_worker_config[0] (local-exec): TASK [kubernetes_worker : Join the node to cluster] ***************************
null_resource.kubernetes_worker_config[0]: Still creating... [2m0s elapsed]
null_resource.kubernetes_worker_config[2]: Still creating... [2m0s elapsed]
null_resource.kubernetes_worker_config[1]: Still creating... [2m0s elapsed]
null_resource.kubernetes_worker_config[2] (local-exec): changed: [192.168.0.173]

null_resource.kubernetes_worker_config[2] (local-exec): PLAY RECAP *********************************************************************
null_resource.kubernetes_worker_config[2] (local-exec): 192.168.0.173            : ok=29   changed=17   unreachable=0    failed=0    skipped=8    rescued=0    ignored=2

null_resource.kubernetes_worker_config[2]: Creation complete after 2m3s [id=4442725935499717348]
null_resource.kubernetes_worker_config[1] (local-exec): changed: [192.168.0.171]

null_resource.kubernetes_worker_config[1] (local-exec): PLAY RECAP *********************************************************************
null_resource.kubernetes_worker_config[1] (local-exec): 192.168.0.171            : ok=29   changed=17   unreachable=0    failed=0    skipped=8    rescued=0    ignored=2

null_resource.kubernetes_worker_config[1]: Creation complete after 2m7s [id=1245252158394522665]
null_resource.kubernetes_worker_config[0]: Still creating... [2m10s elapsed]
null_resource.kubernetes_worker_config[0] (local-exec): changed: [192.168.0.70]

null_resource.kubernetes_worker_config[0] (local-exec): PLAY RECAP *********************************************************************
null_resource.kubernetes_worker_config[0] (local-exec): 192.168.0.70             : ok=29   changed=17   unreachable=0    failed=0    skipped=8    rescued=0    ignored=2

null_resource.kubernetes_worker_config[0]: Creation complete after 2m11s [id=7442378902838305076]

Apply complete! Resources: 18 added, 0 changed, 0 destroyed.
kpora@Krisz-PC:~/kubernetes/terraform_openstack$
```

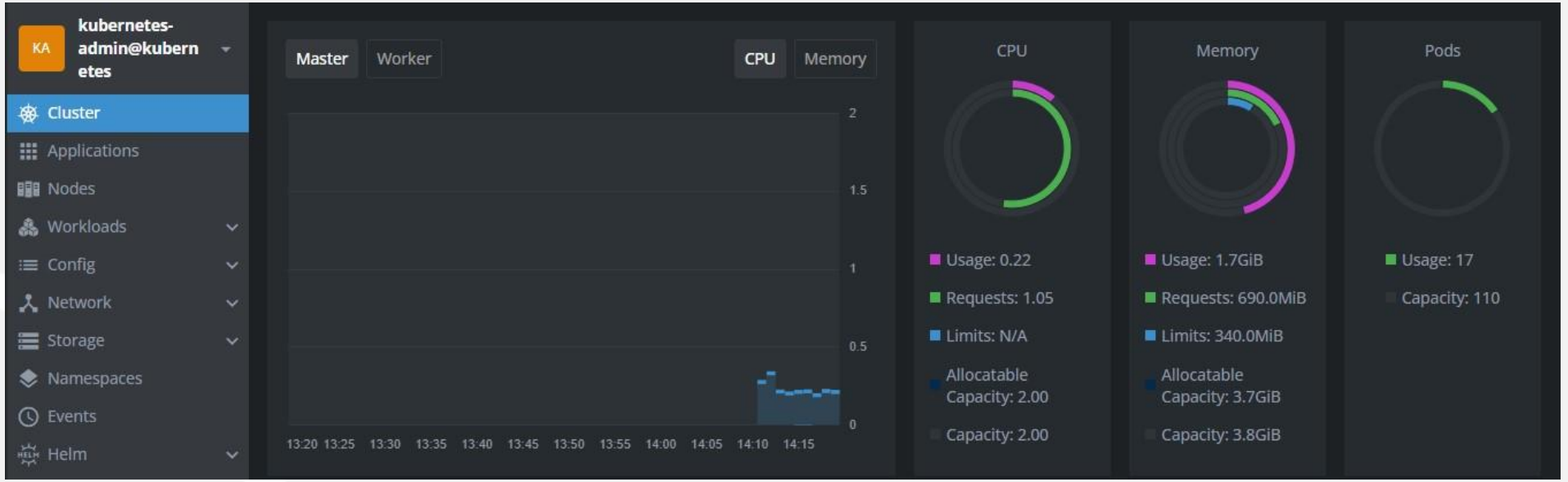
Step 6 - Infrastructure deployment

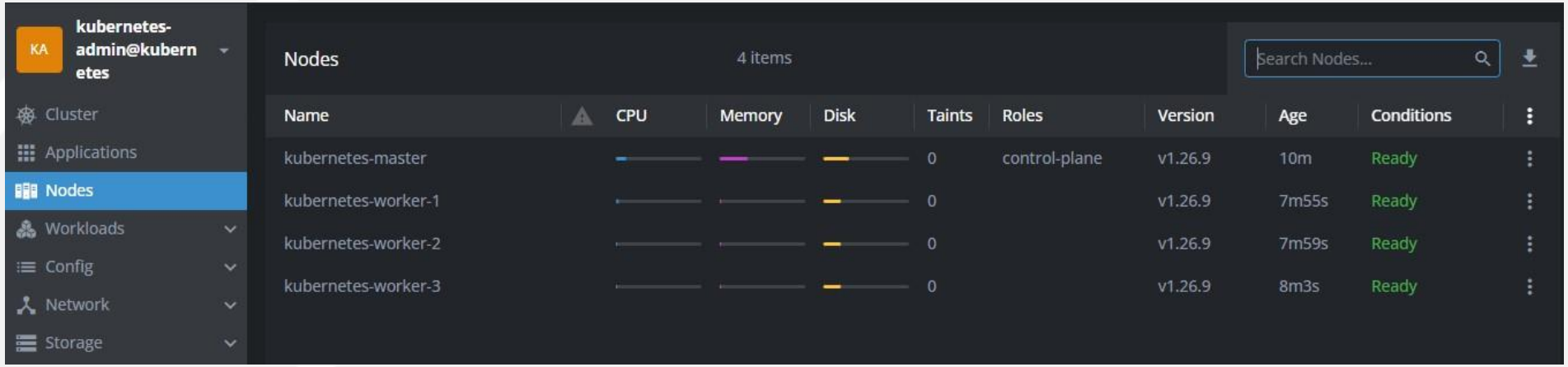


Step 7 - Infrastructure usage - Lens

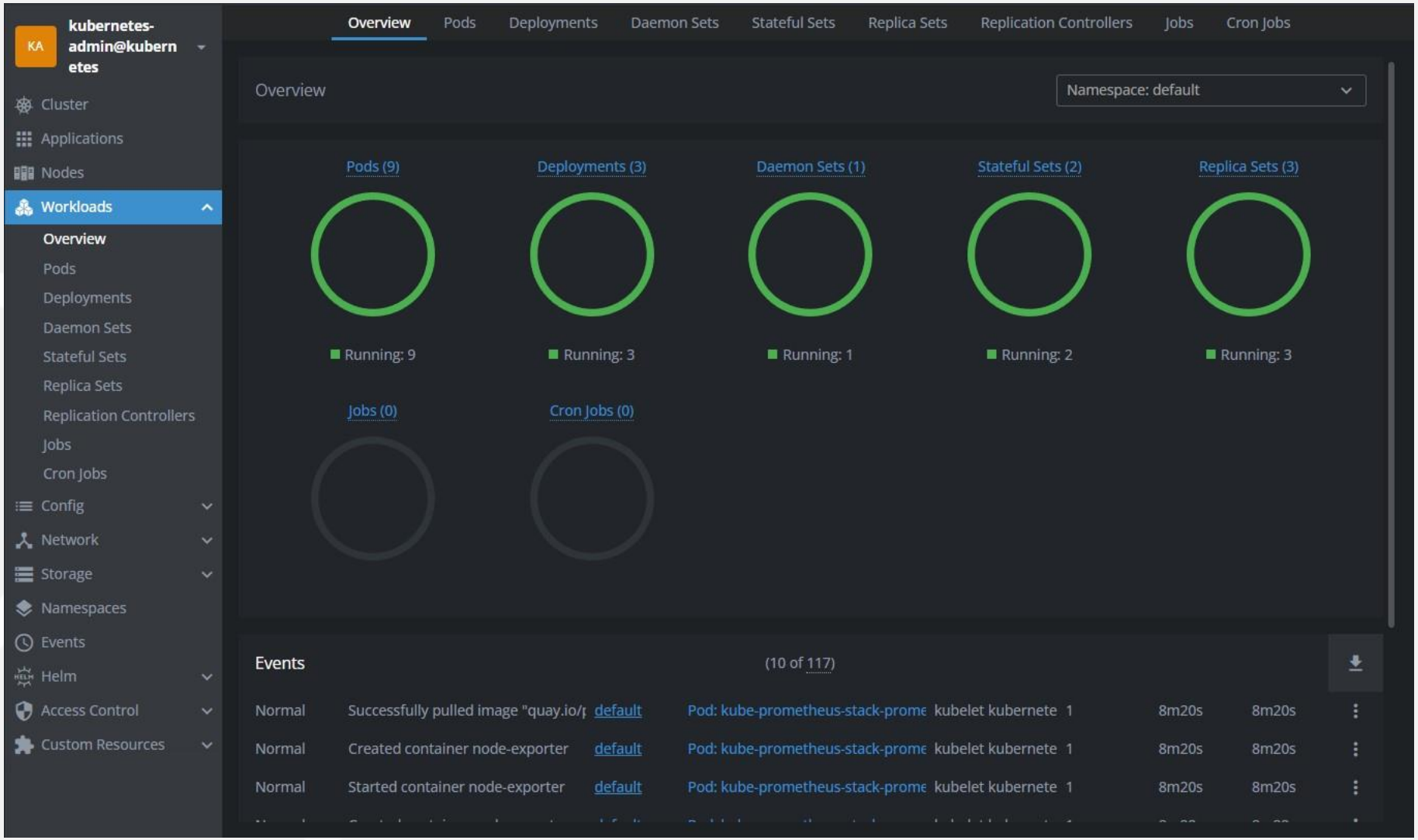


Step 7 - Infrastructure usage - Lens





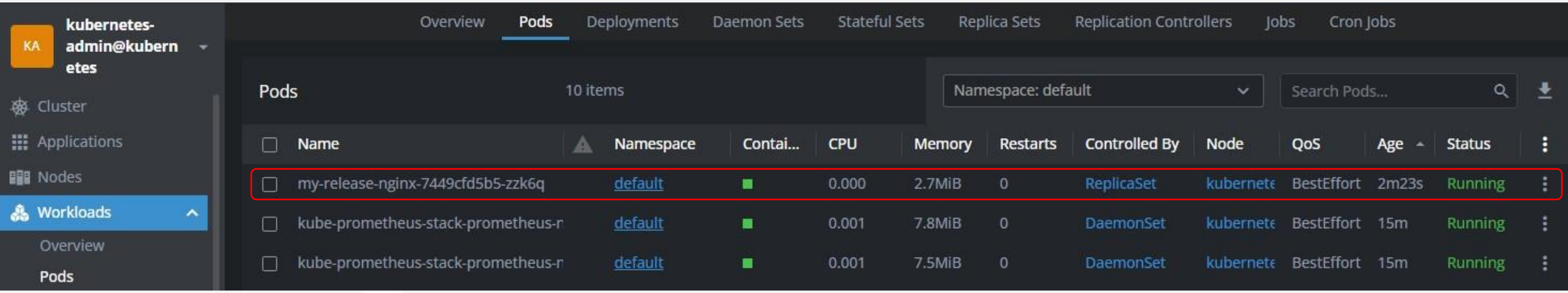
Step 7 - Infrastructure usage - Lens

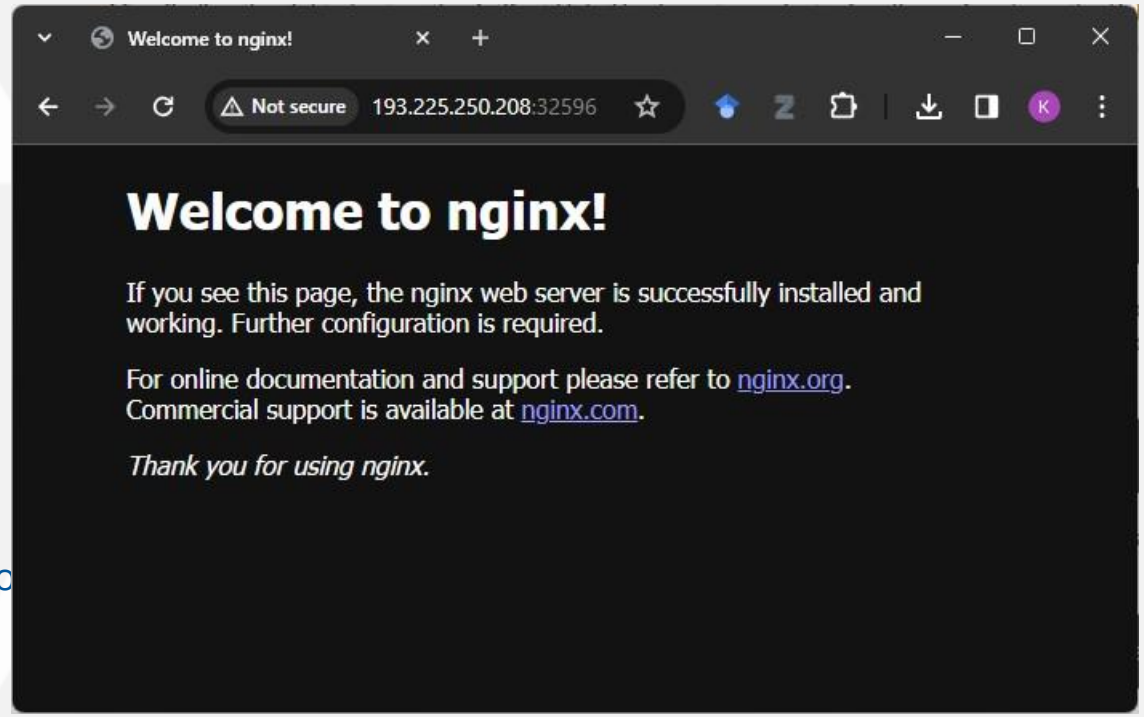


Step 7 - Infrastructure usage - Helm

```
ubuntu@kubernetes-master:~$ helm repo add bitnami https://charts.bitnami.com/bitnami
"bitnami" has been added to your repositories
ubuntu@kubernetes-master:~$ helm install my-release --set service.type=NodePort bitnami/nginx
NAME: my-release
LAST DEPLOYED: Wed Oct 25 12:25:13 2023
NAMESPACE: default
STATUS: deployed
REVISION: 1
TEST SUITE: None
NOTES:
CHART NAME: nginx
CHART VERSION: 15.3.5
APP VERSION: 1.25.3

** Please be patient while the chart is being deployed **
NGINX can be accessed through the following DNS name from within your cluster:

    my-release-nginx.default.svc.cluster.local (port 80)

To access NGINX from outside the cluster, follow the steps below:

1. Get the NGINX URL by running these commands:

    export NODE_PORT=$(kubectl get --namespace default -o jsonpath="{.spec.ports[0].nodePort}" services my-release-nginx)
    export NODE_IP=$(kubectl get nodes --namespace default -o jsonpath="{.items[0].status.addresses[0].address}")
    echo "http://${NODE_IP}:${NODE_PORT}"
ubuntu@kubernetes-master:~$ kubectl get --namespace default -o jsonpath="{.spec.ports[0].nodePort}" services my-release-nginx
32596ubuntu@kubernetes-master:~$
```

Step 7 - Infrastructure usage - Helm



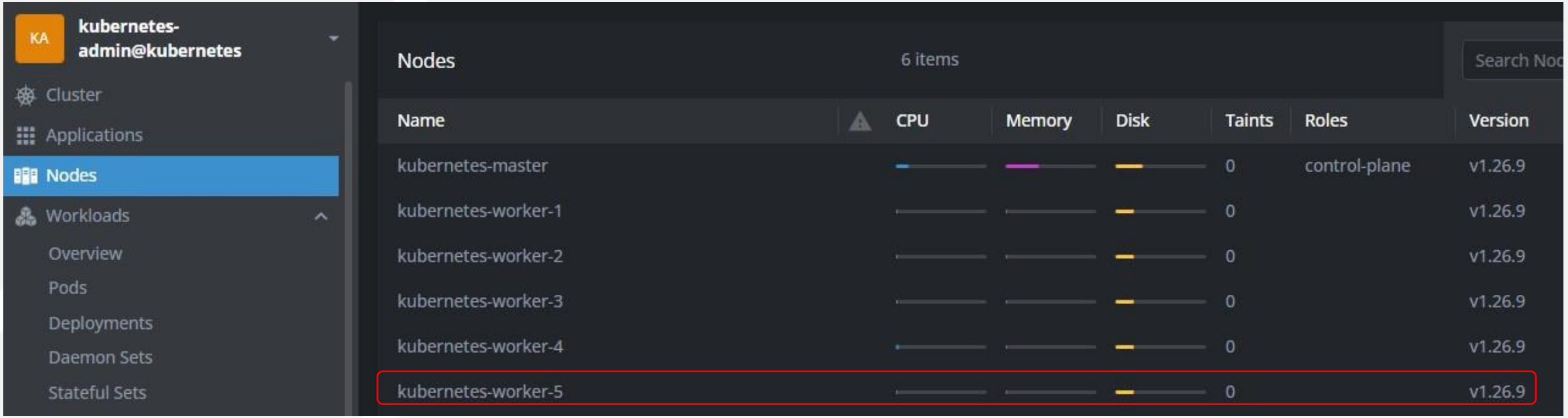


HUN REN | SZTAKI

Step 7 - Infrastructure usage – Scaling

```
kubernetes_worker_node = ({
    name = "kubernetes-worker"
    count = 5
    flavor_name = "m2.medium"
    image_id = "7cf55b87-bd6e-4243-a708-b255bc33df2d"
    key_pair = "kpora"
    volume_size = 30
})
```

Instance Name	Image Name	IP Address	Flavor
kubernetes-master	Ubuntu 22.04 LTS	192.168.0.150, 193.225.250.208	m2.medium
kubernetes-worker-1	Ubuntu 22.04 LTS	192.168.0.70	m2.medium
kubernetes-worker-2	Ubuntu 22.04 LTS	192.168.0.171	m2.medium
kubernetes-worker-3	Ubuntu 22.04 LTS	192.168.0.173	m2.medium
kubernetes-worker-4	Ubuntu 22.04 LTS	192.168.0.59	m2.medium
kubernetes-worker-5	Ubuntu 22.04 LTS	192.168.0.148	m2.medium



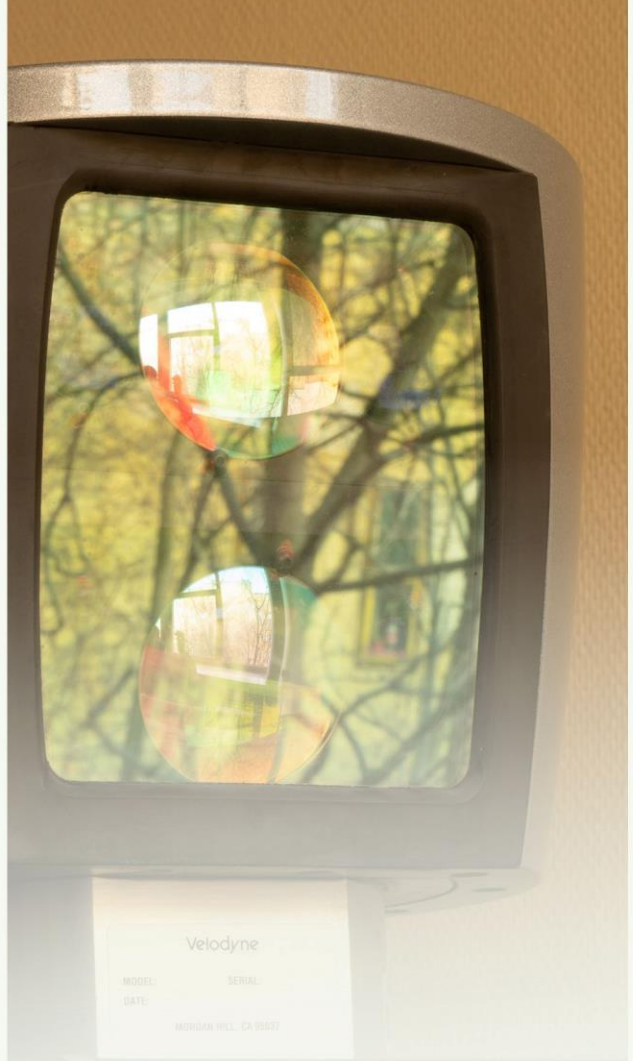
Difference between the SZTAKI and Wigner branches

- A VPN connection is required
  - Cloud API
  - Accessing virtual machines
- Specifying a floating IP (external IP address)
- Adding an insecure parameter to the Provider section
- Connection to the interface remains protected!
  - VPN, Self-signed certificate

```
provider "openstack" {
  auth_url                      = var.auth_data.auth_url
  application_credential_id     = var.auth_data.application_credential_id
  application_credential_secret = var.auth_data.application_credential_secret
  # If your cloud provider uses self-signed cert, use the insecure flag
  # insecure = true
}
```

Summary

- Operation of the Kubernetes reference architecture
- Process of building the reference architecture
- Using Kubernetes
- The creation of different functional Big Data/MI/container/workload manager environments is built automatically using the Terraform tool
- The installation does not require deep IT, network, or software installation and configuration knowledge
- The environments compiled so far (Hadoop, Spark, Tensorflow, Slurm, Horovod, Kafka, Kubernetes, MongoDB, MariaDB, OpenVPN, etc.) are available in the form of a reference architecture and can be tested on the HUN-REN Cloud
- HUN-REN Cloud technical support: info@science-cloud.hu



Thank you for your attention!

HUN REN | SZTAKI