31-1-2022

User manual

Kubernetes with GPU integration

Maenhoudt Tom
STUDENT AT HOWEST

Content

Introduction

This document shows you how to deploy JupyterHub on the Kubernetes cluster with GPUs. If you don't have a working cluster refer to the Installation manual of this project. JupyterHub needs persistent storage to save data, this will be configured with Longhorn. Every node needs open-iscsi. This is preinstalled in Ubuntu but if it is not present you can find more information here

What do you need

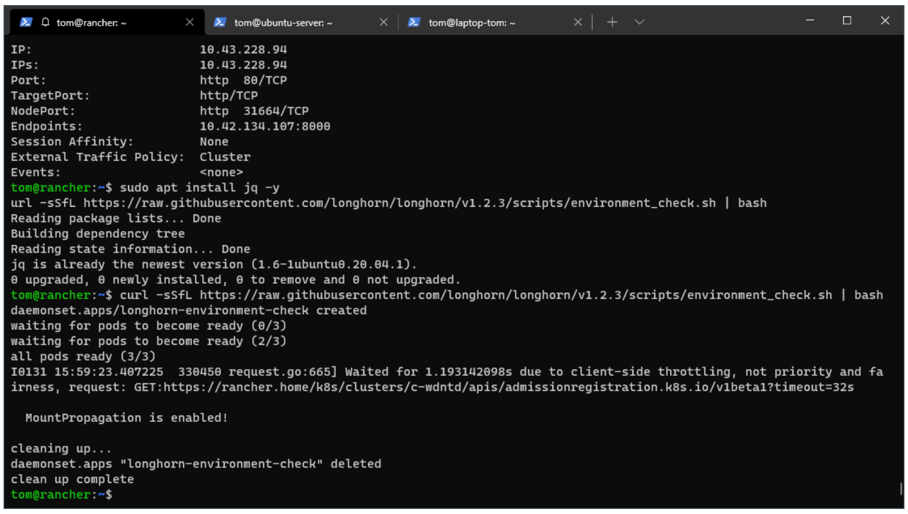
- Working Kubernetes cluster
- Kubectl and Helm
- Access to Rancher

Configure storage, Longhorn

Before installing Longhorn execute the following commands on the master node to check the cluster compatibility.

```
sudo apt install jq -y
curl -sSfL https://raw.githubusercontent.com/longhorn/longhorn/v1.2.3/scripts/environment_check.sh | bash
```

The output should look like this:



```
IP:                       10.43.228.94
IPs:                      10.43.228.94
Port:                     http  80/TCP
TargetPort:               http/TCP
NodePort:                 http  31664/TCP
Endpoints:                10.42.134.107:8000
Session Affinity:         None
External Traffic Policy:  Cluster
Events:                   <none>
tom@rancher:~$ sudo apt install jq -y
url -sSfL https://raw.githubusercontent.com/longhorn/longhorn/v1.2.3/scripts/environment_check.sh | bash
Reading package lists... Done
Building dependency tree
Reading state information... Done
jq is already the newest version (1.6-1ubuntu0.20.04.1).
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
tom@rancher:~$ curl -sSfL https://raw.githubusercontent.com/longhorn/longhorn/v1.2.3/scripts/environment_check.sh | bash
daemonset.apps/longhorn-environment-check created
waiting for pods to become ready (0/3)
waiting for pods to become ready (2/3)
all pods ready (3/3)
I0131 15:59:23.407225  330450 request.go:665] Waited for 1.193142098s due to client-side throttling, not priority and fa
irness, request: GET:https://rancher.home/k8s/clusters/c-wdntd/apis/admissionregistration.k8s.io/v1beta1?timeout=32s

  MountPropagation is enabled!

cleaning up...
daemonset.apps "longhorn-environment-check" deleted
clean up complete
tom@rancher:~$
```

After checking the environment, we will install Longhorn with Helm.

First add the Longhorn repository to Helm and update Helm:

```
helm repo add longhorn https://charts.longhorn.io
helm repo update
```

Install Longhorn with the following command:

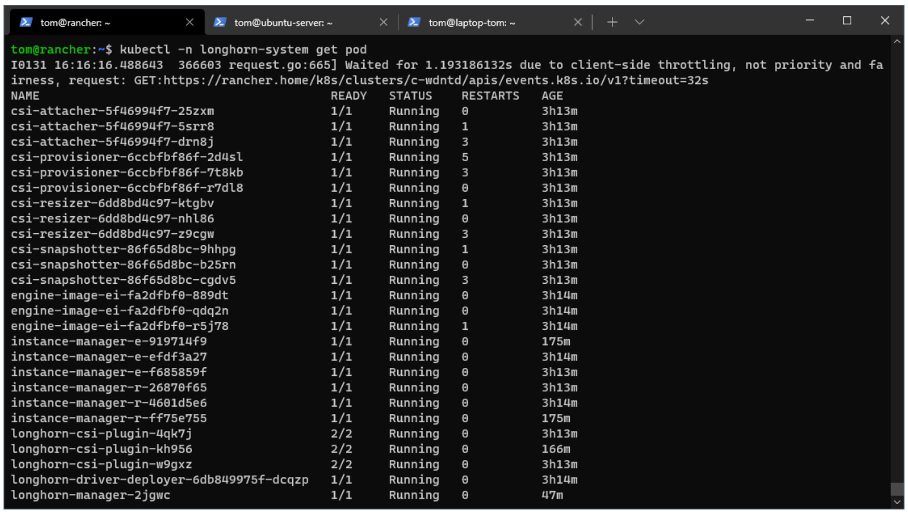
```
helm install longhorn longhorn/longhorn --namespace longhorn-system --create-namespace
```

This command will create a new namespace called longhorn-system. This helps to keep things organized.

To confirm the deployment succeeded execute the following command:

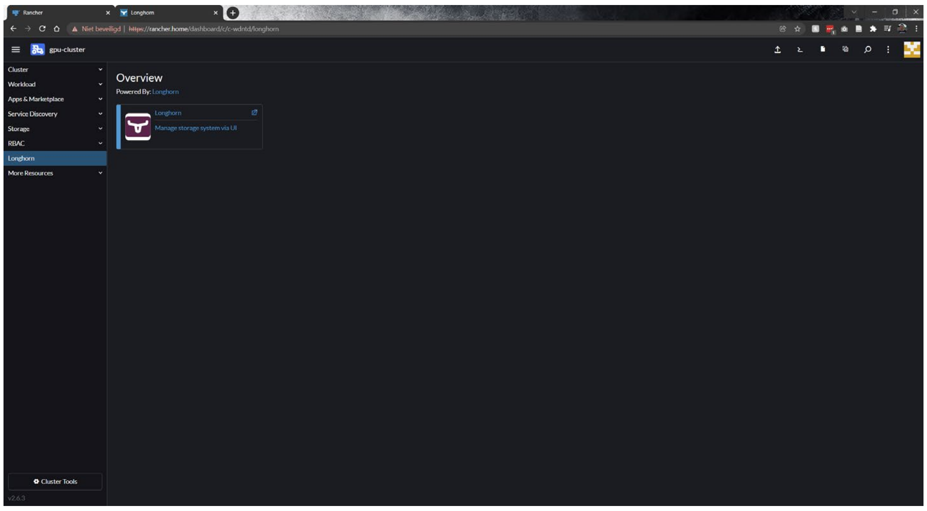
```
kubectl -n longhorn-system get pod
```

The output should look something like this with all the pods running:

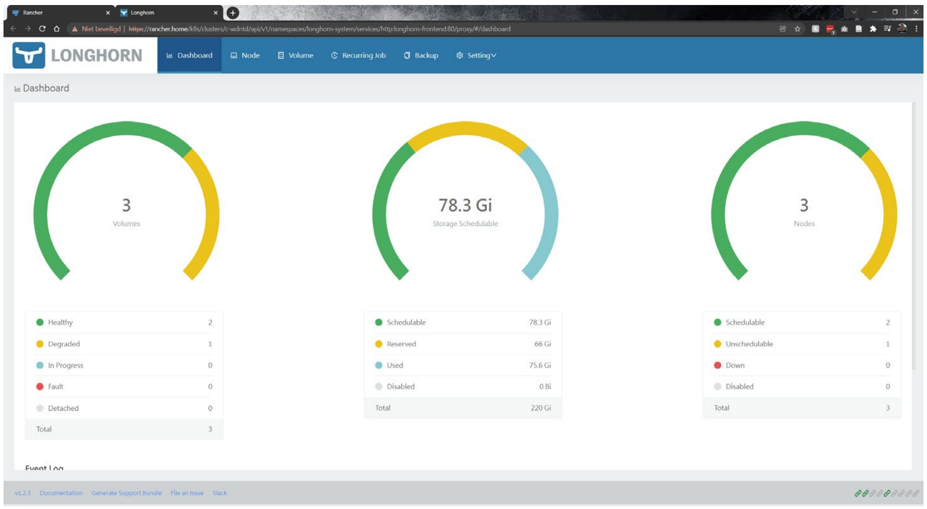


```
tom@rancher:~$ kubectl -n longhorn-system get pod
I0131 16:16:16.488643  366603 request.go:665] Waited for 1.193186132s due to client-side throttling, not priority and fa
irness, request: GET:https://rancher.home/k8s/clusters/c-wdntd/apis/events.k8s.io/v1?timeout=32s
NAME                                          READY   STATUS    RESTARTS   AGE
csi-attacher-5f46994f7-25zxm                  1/1     Running   0          3h13m
csi-attacher-5f46994f7-5srr8                  1/1     Running   1          3h13m
csi-attacher-5f46994f7-drn8j                  1/1     Running   3          3h13m
csi-provisioner-6ccbfbf86f-2d4sl              1/1     Running   5          3h13m
csi-provisioner-6ccbfbf86f-7t8kb              1/1     Running   3          3h13m
csi-provisioner-6ccbfbf86f-r7dl8              1/1     Running   0          3h13m
csi-resizer-6dd8bd4c97-ktgbv                  1/1     Running   1          3h13m
csi-resizer-6dd8bd4c97-nhl86                  1/1     Running   0          3h13m
csi-resizer-6dd8bd4c97-z9cgw                  1/1     Running   3          3h13m
csi-snapshotter-86f65d8bc-9hhpg               1/1     Running   1          3h13m
csi-snapshotter-86f65d8bc-b25rn               1/1     Running   0          3h13m
csi-snapshotter-86f65d8bc-cgdv5               1/1     Running   3          3h13m
engine-image-ei-fa2dfbf0-889dt                1/1     Running   0          3h14m
engine-image-ei-fa2dfbf0-qdq2n                1/1     Running   0          3h14m
engine-image-ei-fa2dfbf0-r5j78                1/1     Running   1          3h14m
instance-manager-e-919714f9                   1/1     Running   0          175m
instance-manager-e-efdf3a27                   1/1     Running   0          3h14m
instance-manager-e-f685859f                   1/1     Running   0          3h13m
instance-manager-r-26870f65                   1/1     Running   0          3h13m
instance-manager-r-4601d5e6                   1/1     Running   0          3h14m
instance-manager-r-ff75e755                   1/1     Running   0          175m
longhorn-csi-plugin-4qk7j                     2/2     Running   0          3h13m
longhorn-csi-plugin-kh956                     2/2     Running   0          166m
longhorn-csi-plugin-w9gxz                     2/2     Running   0          3h13m
longhorn-driver-deployer-6db849975f-dcqzp     1/1     Running   0          3h14m
longhorn-manager-2jgwc                        1/1     Running   0          47m
```

You can access the Longhorn UI through Rancher. Refresh the Rancher UI and on the overview page of the cluster you should see a new option on the left side called Longhorn.



When you click on the Longhorn icon you will be redirected to the Longhorn GUI.

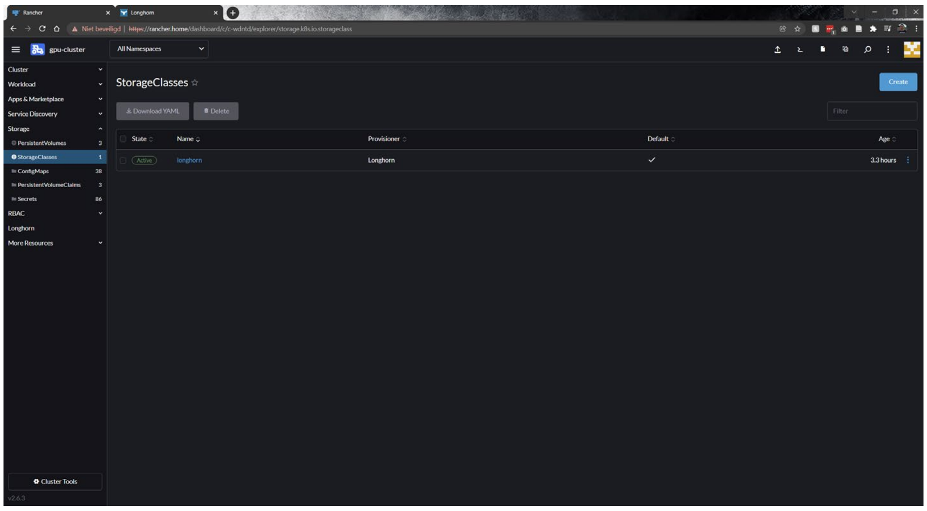


● Healthy	2
● Degraded	1
● In Progress	0
● Fault	0
● Detached	0
Total	3

● Schedulable	78.3 Gi
● Reserved	66 Gi
● Used	75.6 Gi
● Disabled	0 Bi
Total	220 Gi

● Schedulable	2
● Unschedulable	1
● Down	0
● Disabled	0
Total	3

In the Rancher UI under Storage and Storage Classes you will see that a Longhorn storage class is created. This will dynamically provision storage for the apps. This storage is persistent for all the nodes.



State	Name	Provisioner	Default	Age
Active	longhorn	Longhorn	✓	3.3 hours

Now that the storage is set up JupyterHub can be installed.

Install JupyterHub

To install JupyterHub you need to create a Helm config file first. This file is used to customize the Helm chart to suit your needs. For now, this can be empty or with some helpful comments.

Create config.yaml with the following helpful comments:

```
# This file can update the JupyterHub Helm chart's default configuration values.
#
# For reference see the configuration reference and default values, but make
# sure to refer to the Helm chart version of interest to you!
#
# Introduction to YAML:     https://www.youtube.com/watch?v=cdLNKUoMc6c
# Chart config reference:   https://zero-to-jupyterhub.readthedocs.io/en/stable/resources/reference.html
# Chart default values:     https://github.com/jupyterhub/zero-to-jupyterhub-k8s/blob/HEAD/jupyterhub/values.yaml
# Available chart versions: https://jupyterhub.github.io/helm-chart/
#
```

If you want a demo notebook with GPUs assigned to it you can use the following content for the config.yaml file:

```
hub:
  image:
    tag: 0.11.1-n270.he08bb7f8
  extraConfig:
    runaiConfig: |
      c.KubeSpawner.scheduler_name = u'runai-scheduler'
      c.KubeSpawner.extra_labels = {
          'project': '{username}',
          'user': '{username}',
          'priorityClassName': 'build'
      }
      c.KubeSpawner.environment = {
          'JUPYTERHUB_API_URL': 'http://hub.jhub.svc.cluster.local:8081/hub/api'
      }
      c.KubeSpawner.enable_user_namespaces = True
      c.KubeSpawner.user_namespace_template = u'runai-{username}'
      def pre_spawn_hook(spawner):
          spawner.log.debug("pre_spawn_hook start")
          async def dummy_ensure_namespace():
              spawner.log.debug("dummy ensure_namespace")
          spawner._ensure_namespace = dummy_ensure_namespace
          spawner.log.debug("pre_spawn_hook end")
      c.Spawner.pre_spawn_hook = pre_spawn_hook
proxy:
  secretToken: <SECRET-TOKEN>
singleuser:
  storage:
    type: none
  profileList:
  - default: true
    slug: "RA 1"
    description: "Run:AI Profile"
    display_name: "Run:AI with 1 GPU"
    kubespawner_override:
      image: jupyter/datascience-notebook
      tag: 177037d09156
      extra_resource_limits:
        nvidia.com/gpu: "1"
  - slug: "RA 2"
    description: "Run:AI Profile"
    display_name: "Run:AI with 2 GPU"
    kubespawner_override:
      image: jupyter/tensorflow-notebook
      tag: 3395de4db93a
      extra_resource_limits:
        nvidia.com/gpu: "2"
```

replace secret-token with the output of the following command:

```
openssl rand -hex 32
```

Next add the JupyterHub repo to Helm and update Helm:

```
helm repo add jupyterhub https://jupyterhub.github.io/helm-chart/
helm repo update
```

Next run the following command to install JupyterHub:

```
helm upgrade --cleanup-on-fail \
  --install <helm-release-name> jupyterhub/jupyterhub \
  --namespace <k8s-namespace> \
  --create-namespace \
  --version=<chart-version> \
  --values config.yaml
```

The Helm release name and namespace can be whatever makes sense for you. This is to keep things organized. The version corresponds to the version of the Helm chart. A list of the versions can be found here.

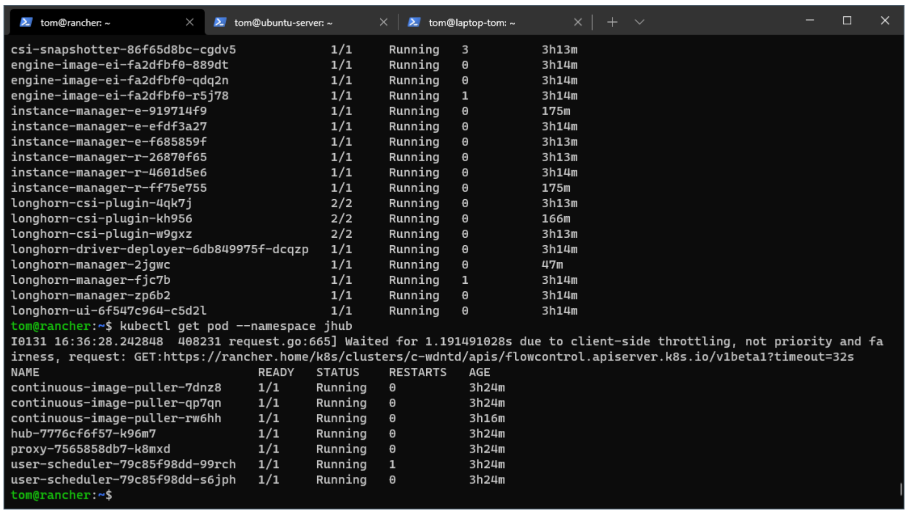
This is the command that I use to install JupyterHub:

```
helm upgrade --cleanup-on-fail \
  --install jhub jupyterhub/jupyterhub \
  --namespace jhub \
  --create-namespace \
  --version=1.2.0 \
  --values config.yaml
```

Once this step is finished you can check the status of the pods with the following command. All the pods should be running. Change the namespace to what you chose.

```
kubectl get pod --namespace <k8s-namespace>
```

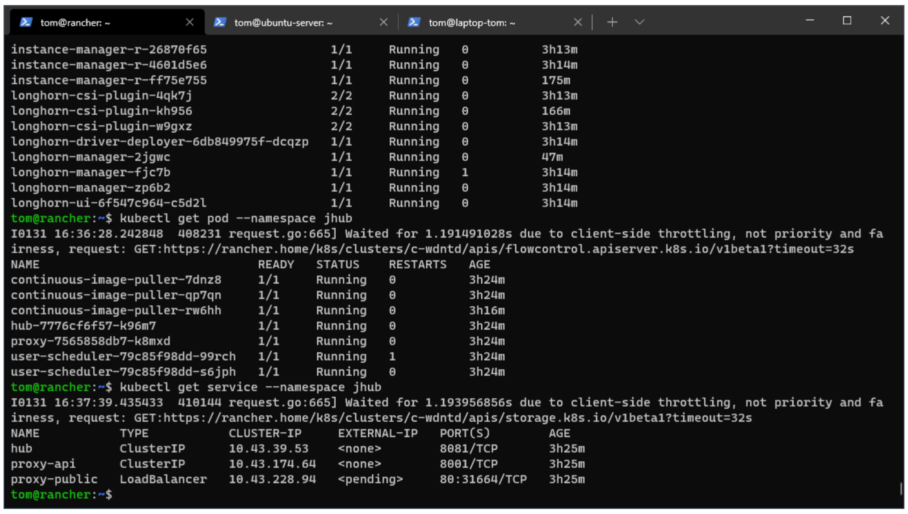
The output should look like this:



```
csi-snapshotter-86f65d8bc-cgdv5            1/1     Running   3          3h13m
engine-image-ei-fa2dfbf0-889dt             1/1     Running   0          3h14m
engine-image-ei-fa2dfbf0-qdq2n             1/1     Running   0          3h14m
engine-image-ei-fa2dfbf0-r5j78             1/1     Running   1          3h14m
instance-manager-e-919714f9                1/1     Running   0          175m
instance-manager-e-efdf3a27                1/1     Running   0          3h14m
instance-manager-e-f685859f                1/1     Running   0          3h13m
instance-manager-r-26870f65                1/1     Running   0          3h13m
instance-manager-r-4601d5e6                1/1     Running   0          3h14m
instance-manager-r-ff75e755                1/1     Running   0          175m
longhorn-csi-plugin-4qk7j                  2/2     Running   0          3h13m
longhorn-csi-plugin-kh956                  2/2     Running   0          166m
longhorn-csi-plugin-w9gxz                  2/2     Running   0          3h13m
longhorn-driver-deployer-6db849975f-dcqzp  1/1     Running   0          3h14m
longhorn-manager-2jgwc                     1/1     Running   0          47m
longhorn-manager-fjc7b                     1/1     Running   1          3h14m
longhorn-manager-zp6b2                     1/1     Running   0          3h14m
longhorn-ui-6f547c964-c5d2l                1/1     Running   0          3h14m
tom@rancher:~$ kubectl get pod --namespace jhub
I0131 16:36:28.242848  408231 request.go:665] Waited for 1.191491028s due to client-side throttling, not priority and fa
irness, request: GET:https://rancher.home/k8s/clusters/c-wdntd/apis/flowcontrol.apiserver.k8s.io/v1beta1?timeout=32s
NAME                                READY   STATUS    RESTARTS   AGE
continuous-image-puller-7dnz8       1/1     Running   0          3h24m
continuous-image-puller-qp7qn       1/1     Running   0          3h24m
continuous-image-puller-rw6hh       1/1     Running   0          3h16m
hub-7776cf6f57-k96m7                1/1     Running   0          3h24m
proxy-7565858db7-k8mxd              1/1     Running   0          3h24m
user-scheduler-79c85f98dd-99rch     1/1     Running   1          3h24m
user-scheduler-79c85f98dd-s6jph     1/1     Running   0          3h24m
tom@rancher:~$
```

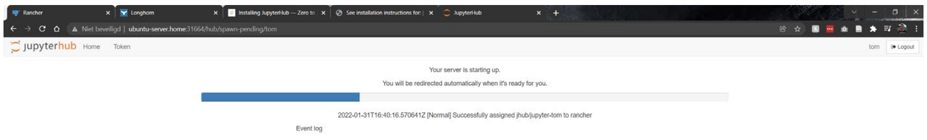
To access JupyterHub run the following command to get the IP-address and port.

```
kubectl get service --namespace <k8s-namespace>
```



```
instance-manager-r-26870f65                 1/1     Running   0          3h13m
instance-manager-r-4601d5e6                 1/1     Running   0          3h14m
instance-manager-r-ff75e755                 1/1     Running   0          175m
longhorn-csi-plugin-4qk7j                   2/2     Running   0          3h13m
longhorn-csi-plugin-kh956                   2/2     Running   0          166m
longhorn-csi-plugin-w9gxz                   2/2     Running   0          3h13m
longhorn-driver-deployer-6db849975f-dcqzp   1/1     Running   0          3h14m
longhorn-manager-2jgwc                      1/1     Running   0          47m
longhorn-manager-fjc7b                      1/1     Running   1          3h14m
longhorn-manager-zp6b2                      1/1     Running   0          3h14m
longhorn-ui-6f547c964-c5d2l                 1/1     Running   0          3h14m
tom@rancher:~$ kubectl get pod --namespace jhub
I0131 16:36:28.242848  408231 request.go:665] Waited for 1.191491028s due to client-side throttling, not priority and fa
irness, request: GET:https://rancher.home/k8s/clusters/c-wdntd/apis/flowcontrol.apiserver.k8s.io/v1beta1?timeout=32s
NAME                                 READY   STATUS    RESTARTS   AGE
continuous-image-puller-7dnz8        1/1     Running   0          3h24m
continuous-image-puller-qp7qn        1/1     Running   0          3h24m
continuous-image-puller-rw6hh        1/1     Running   0          3h16m
hub-7776cf6f57-k96m7                 1/1     Running   0          3h24m
proxy-7565858db7-k8mxd               1/1     Running   0          3h24m
user-scheduler-79c85f98dd-99rch      1/1     Running   1          3h24m
user-scheduler-79c85f98dd-s6jph      1/1     Running   0          3h24m
tom@rancher:~$ kubectl get service --namespace jhub
I0131 16:37:39.435433  410144 request.go:665] Waited for 1.193956856s due to client-side throttling, not priority and fa
irness, request: GET:https://rancher.home/k8s/clusters/c-wdntd/apis/storage.k8s.io/v1beta1?timeout=32s
NAME           TYPE           CLUSTER-IP      EXTERNAL-IP   PORT(S)        AGE
hub            ClusterIP      10.43.39.53     <none>        8081/TCP       3h25m
proxy-api      ClusterIP      10.43.174.64    <none>        8001/TCP       3h25m
proxy-public   LoadBalancer   10.43.228.94    <pending>     80:31664/TCP   3h25m
tom@rancher:~$
```

You can access JupyterHub by going to your browser and filling in the IP-address and port. If the Load balancer has no external IP-address you take the IP-address of one of the nodes.


Rancher
Longhorn
Installing JupyterHub — Zero to
See installation instructions for
JupyterHub
Niet beveiligd | ubuntu-server.home:31664/hub/spawn-pending/tom
jupyterhub
Home
Token
tom
Logout
Event log

Your server is starting up.

You will be redirected automatically when it's ready for you.

2022-01-31T16:40:16.570641Z [Normal] Successfully assigned jhub/jupyter-tom to rancher

The server will start, and you can log in with any user and password combination. The storage is created based on the username. See the documentation to customize the installation to suit your needs.

Now you can run Jupyter notebooks on the cluster and take advantage of the GPUs.

Sources

[1]

"Installing JupyterHub — Zero to JupyterHub with Kubernetes documentation." https://zero-to-jupyterhub.readthedocs.io/en/latest/jupyterhub/installation.html (accessed Jan. 31, 2022).

[2]

"Longhorn | The Longhorn Documentation," *Longhorn*. https://longhorn.io/docs/1.2.3/ (accessed Jan. 31, 2022).

[3]

"See installation instructions for:," *JupyterHub's Helm chart repository*. https://jupyterhub.github.io/helm-chart/ (accessed Jan. 31, 2022).