Installing microk8s and Kubeflow on the Server

If the server is connected to LAN inaccessible from outside we can disable the ufw firewall by issuing the following command on the server:

```
sudo ufw disable
```

Otherwise keep the firewall running and add appropriate firewall rule. For details consult this link.

We are going to use **microk8s** for a single node deployment with GPU support on the server which we assembled. The beauty of microk8s is that it runs directly on the Linux server hardware without the help of virtual machine (e.g. minikube). As such enabling GPU support for microk8s is trivial and provides important speedup with ML workloads.

We are going to install microk8s v1.20 via snap:

```
sudo snap install microk8s --classic --channel=1.20/stable
```

For details visit the official microk8s docs on this link and this link.

After about 10 mins the basic microk8s installation completes.

We manually enable the following add-ons:

```
microk8s enable gpu
```

```
microk8s dns ingress dashboard
```

and finally

```
microk8s enable kubeflow.
```

Notice the output of the last command:

```
Congratulations, Kubeflow is now available.

The dashboard is available at http://10.64.140.43.xip.io

    Username: admin
    Password: VZWQ**********************S3

To see these values again, run:

    microk8s juju config dex-auth static-username
    microk8s juju config dex-auth static-password


To tear down Kubeflow and associated infrastructure, run:

    microk8s disable kubeflow
```

We are going to expose the Kubernetes dashboard running on our server to a remote host on the same LAN.

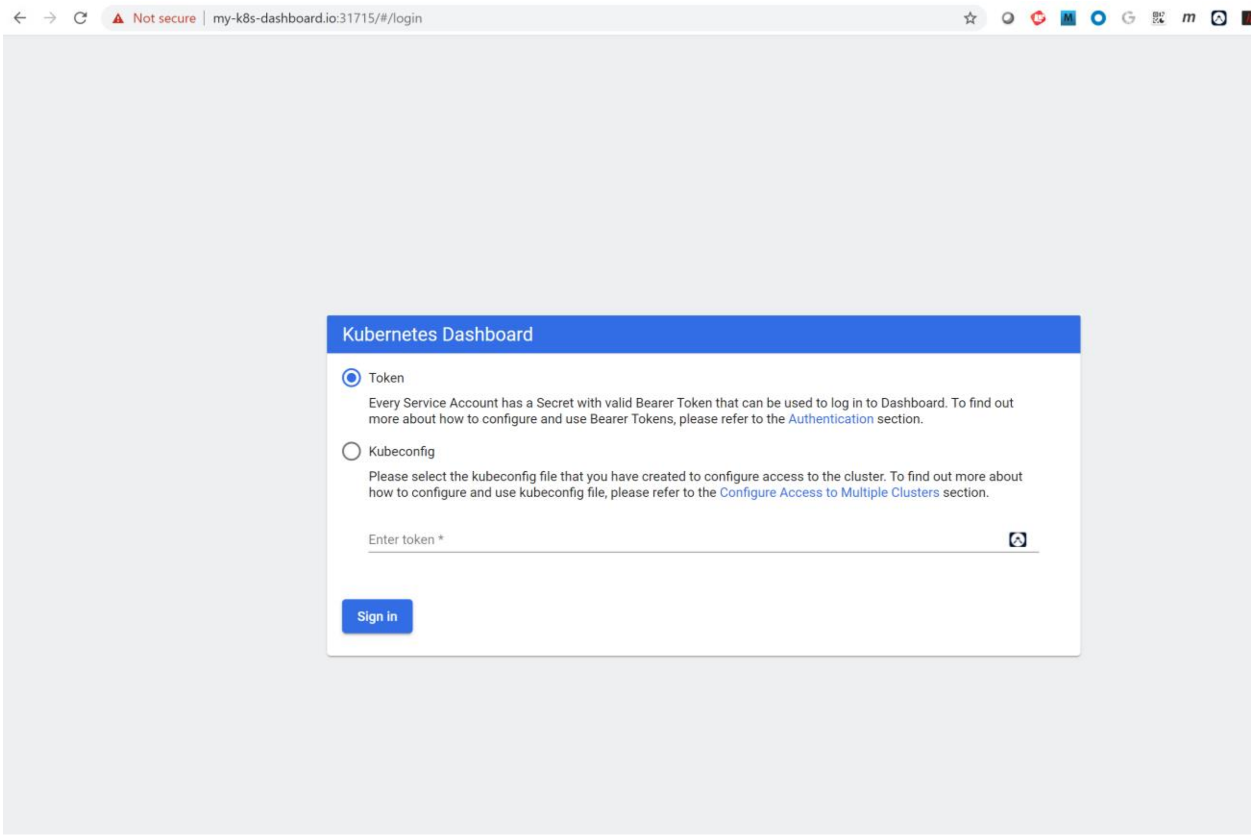
To make our life complicated let's assume our remote host runs Windows 10. Open its hosts file located in C:\Windows\System32\drivers\etc and add the following entry at the bottom:

```
<my-server-ip>   my-k8s-dashboard.io
```

Then on the server find the string value of default token with the lines below:

```
token=$(microk8s kubectl -n kube-system get secret | grep default-token | cut -d " " -f1)
microk8s kubectl -n kube-system describe secret $token
```

The token is sent to **stdout**. Copy the token and paste it below on a browser on our remote host:



Not secure | my-k8s-dashboard.io:31715/#/login

Kubernetes Dashboard

Token

Every Service Account has a Secret with valid Bearer Token that can be used to log in to Dashboard. To find out more about how to configure and use Bearer Tokens, please refer to the Authentication section.

Kubeconfig

Please select the kubeconfig file that you have created to configure access to the cluster. To find out more about how to configure and use kubeconfig file, please refer to the Configure Access to Multiple Clusters section.

Enter token *

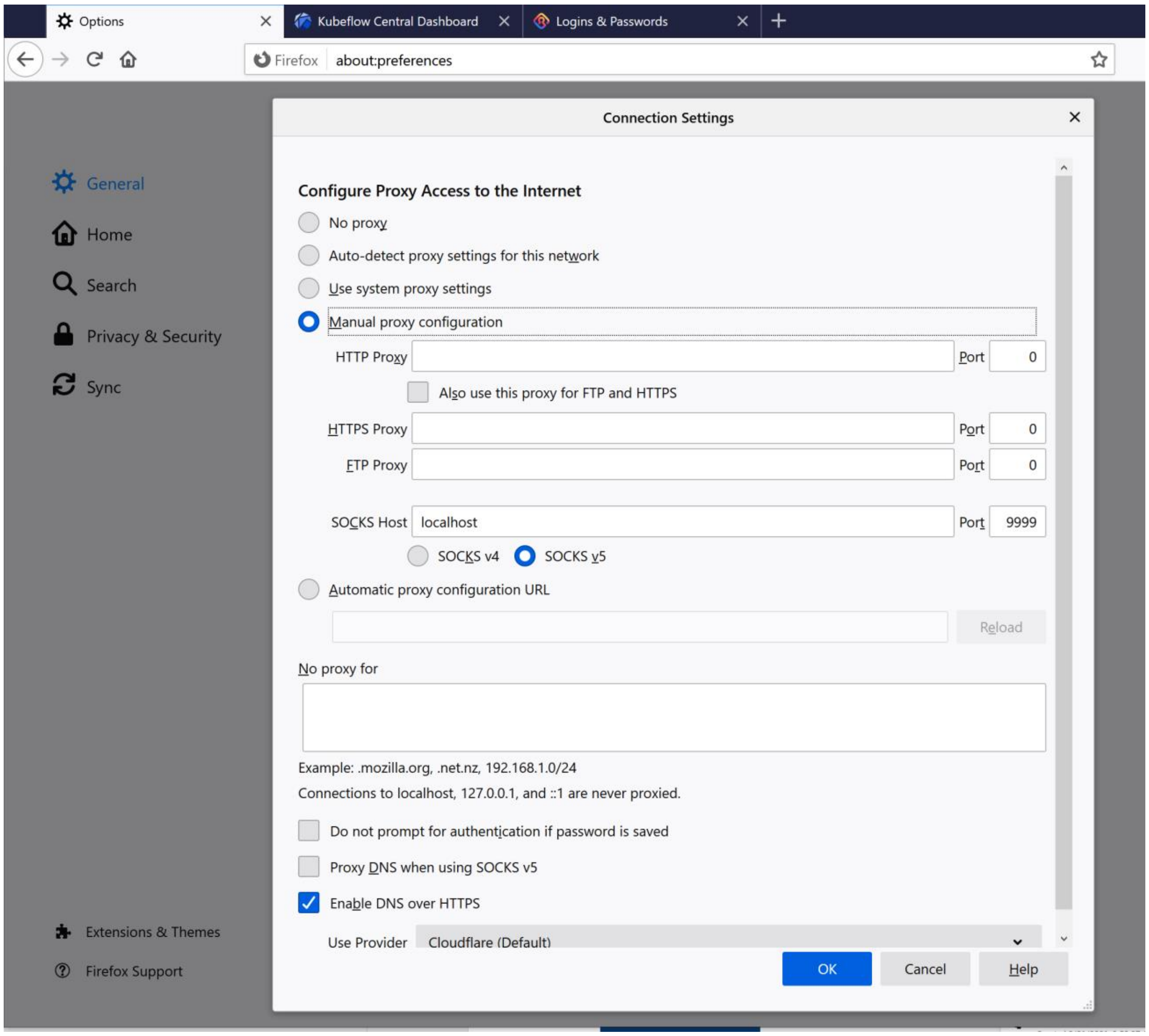
Sign in

We are going to use SOCKS5 proxy for forwarding our requests to the server:

For details see this link and this github microk8s thread.

```
ssh -D9999 <username>@<network-server-ip>
```

We are going to configure Mozilla browser to use SOCKS5 Proxy Access to internet on port 9999 as shown below:



Connection Settings

Configure Proxy Access to the Internet

No proxy

Auto-detect proxy settings for this network

Use system proxy settings

[✓ Manual proxy configuration]

HTTP Proxy | Port 0

Also use this proxy for FTP and HTTPS

HTTPS Proxy | Port 0

FTP Proxy | Port 0

SOCKS Host localhost | Port 9999

SOCKS v4 [✓ SOCKS v5]

Automatic proxy configuration URL

Reload

No proxy for

Example: .mozilla.org, .net.nz, 192.168.1.0/24

Connections to localhost, 127.0.0.1, and ::1 are never proxied.

Do not prompt for authentication if password is saved

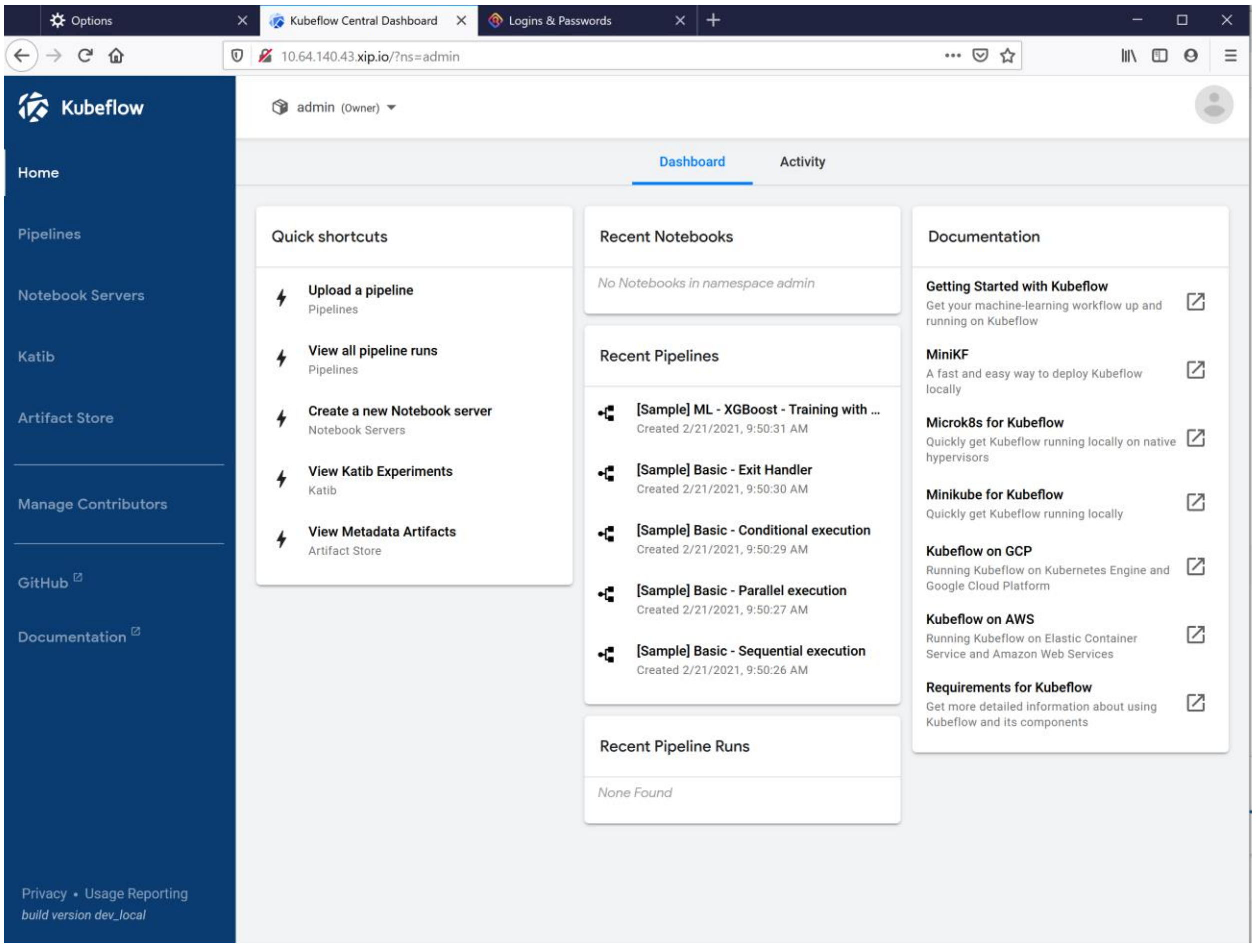
Proxy DNS when using SOCKS v5

[✓ Enable DNS over HTTPS]

Use Provider Cloudflare (Default)

OK Cancel Help

Then we can access the Kubeflow dashboard


Options
Kubeflow Central Dashboard
Logins & Passwords

10.64.140.43.xip.io/?ns=admin

Kubeflow

admin (Owner)

Home
Pipelines
Notebook Servers
Katib
Artifact Store
Manage Contributors
GitHub
Documentation

Privacy • Usage Reporting
build version dev_local

Dashboard
Activity

Quick shortcuts

Upload a pipeline
Pipelines

View all pipeline runs
Pipelines

Create a new Notebook server
Notebook Servers

View Katib Experiments
Katib

View Metadata Artifacts
Artifact Store

Recent Notebooks

No Notebooks in namespace admin

Recent Pipelines

[Sample] ML - XGBoost - Training with ...
Created 2/21/2021, 9:50:31 AM

[Sample] Basic - Exit Handler
Created 2/21/2021, 9:50:30 AM

[Sample] Basic - Conditional execution
Created 2/21/2021, 9:50:29 AM

[Sample] Basic - Parallel execution
Created 2/21/2021, 9:50:27 AM

[Sample] Basic - Sequential execution
Created 2/21/2021, 9:50:26 AM

Recent Pipeline Runs

None Found

Documentation

Getting Started with Kubeflow
Get your machine-learning workflow up and running on Kubeflow

MiniKF
A fast and easy way to deploy Kubeflow locally

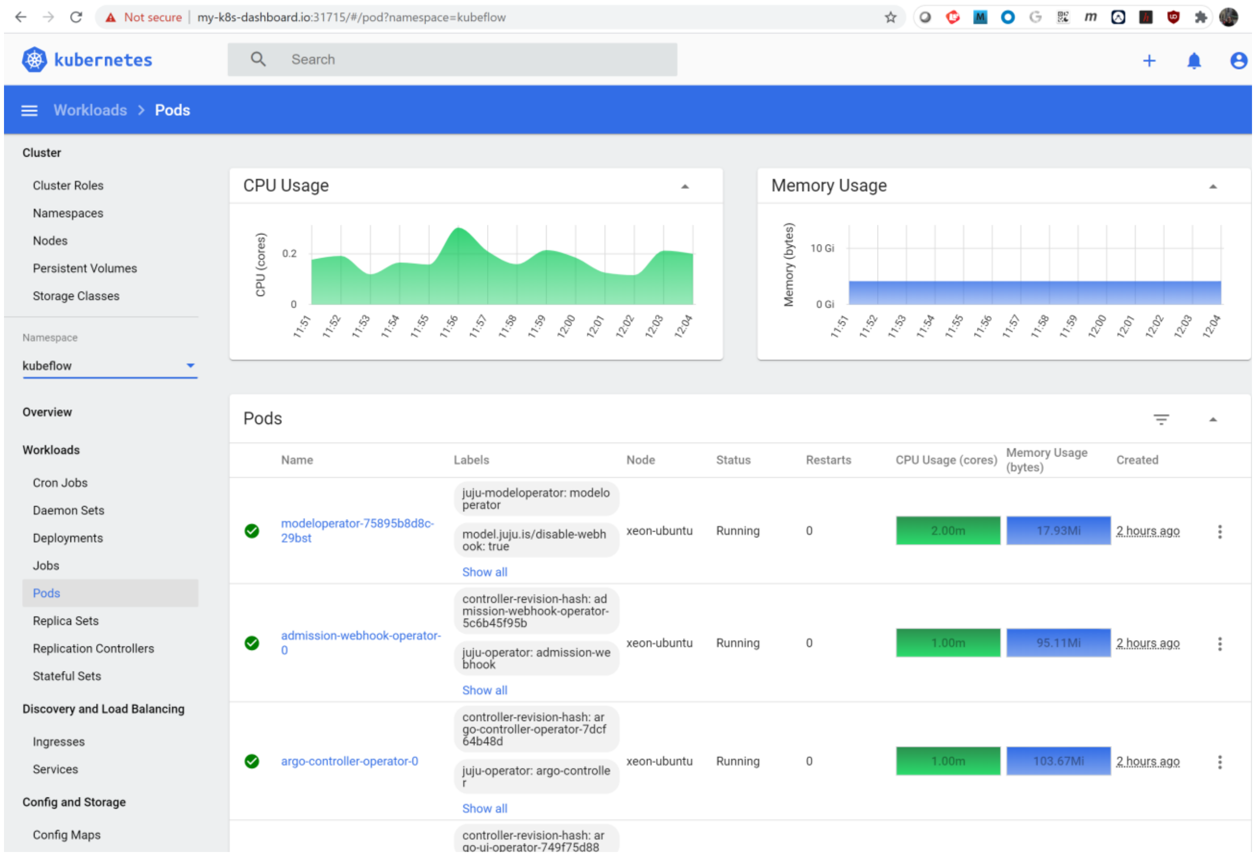
Microk8s for Kubeflow
Quickly get Kubeflow running locally on native hypervisors

Minikube for Kubeflow
Quickly get Kubeflow running locally

Kubeflow on GCP
Running Kubeflow on Kubernetes Engine and Google Cloud Platform

Kubeflow on AWS
Running Kubeflow on Elastic Container Service and Amazon Web Services

Requirements for Kubeflow
Get more detailed information about using Kubeflow and its components



Pods

	Name	Labels	Node	Status	Restarts	CPU Usage (cores)	Memory Usage (bytes)	Created
✓	modeloperator-75895b8d8c-29bst	juju-modeloperator: modeloperator; model.juju.is/disable-webhook: true; Show all	xeon-ubuntu	Running	0	2.00m	17.93Mi	2 hours ago
✓	admission-webhook-operator-0	controller-revision-hash: admission-webhook-operator-5c6b45f95b; juju-operator: admission-webhook; Show all	xeon-ubuntu	Running	0	1.00m	95.11Mi	2 hours ago
✓	argo-controller-operator-0	controller-revision-hash: argo-controller-operator-7dcf64b48d; juju-operator: argo-controller; Show all	xeon-ubuntu	Running	0	1.00m	103.67Mi	2 hours ago
		controller-revision-hash: argo-ui-operator-749f75d88						