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](https://www.digitalocean.com/community/tutorials/ufw-essentials-common-firewall-rules-and-commands).

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](https://microk8s.io/docs) and [this link](https://microk8s.io/docs/setting-snap-channel).

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:

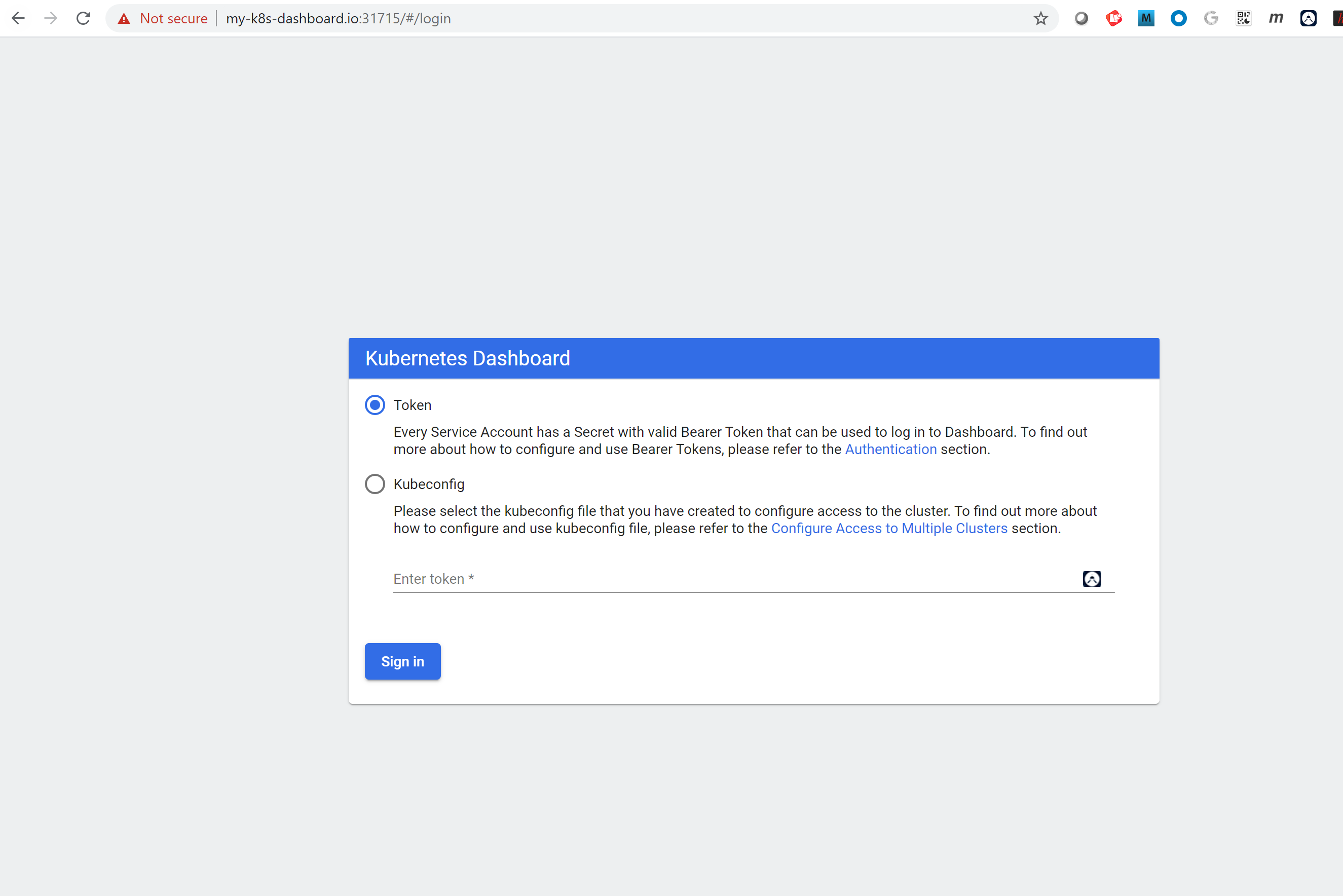
**<my-server-ip> my-k8s-dashboard.io**

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:

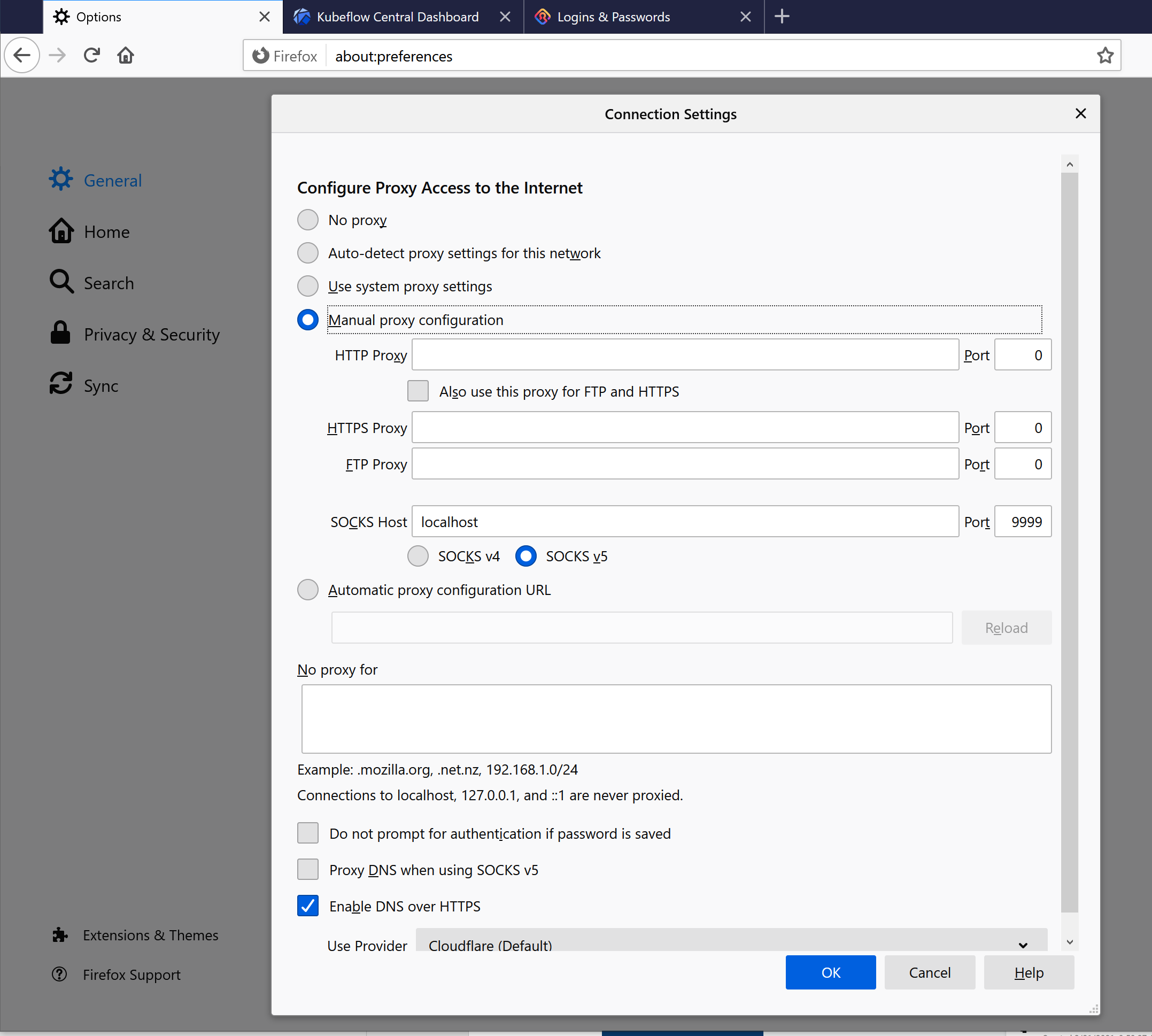


We are going to use SOCKS5 proxy for forwarding our requests to the server:

For details see [this link](https://charmed-kubeflow.io/docs/dashboard) and [this github microk8s thread](https://github.com/ubuntu/microk8s/issues/1243).

**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:



Then we can access the Kubeflow dashboard

