OpenFaaS Platform Usage

Installation and set up:

1. *Install k3s on Master node*
2. *Install k3s on Worker nodes*
3. *Install OpenfaaS on Master node*
4. *Install faas-cli on Master node*

Function Deployment:

1. *Install mqtt-connector*
2. *Build or create the function we want to deploy*

Installation and set up

1. Install k3s on Master node

Via script:

curl -sfL https://get.k3s.io | sh

```
barbara@kube-master:~$ curl -sfL https://get.k3s.io | sh -
[sudo] password for barbara:
[INFO]  Finding release for channel stable
[INFO]  Using v1.25.3+k3s1 as release
[INFO]  Downloading hash https://github.com/k3s-io/k3s/releases/download/v1.25.3+k3s1/sha256sum-amd64.txt
[INFO]  Downloading binary https://github.com/k3s-io/k3s/releases/download/v1.25.3+k3s1/k3s
[INFO]  Verifying binary download
[INFO]  Installing k3s to /usr/local/bin/k3s
[INFO]  Skipping installation of SELinux RPM
[INFO]  Creating /usr/local/bin/kubectl symlink to k3s
[INFO]  Creating /usr/local/bin/crictl symlink to k3s
[INFO]  Creating /usr/local/bin/ctr symlink to k3s
[INFO]  Creating killall script /usr/local/bin/k3s-killall.sh
[INFO]  Creating uninstall script /usr/local/bin/k3s-uninstall.sh
[INFO]  env: Creating environment file /etc/systemd/system/k3s.service.env
[INFO]  systemd: Creating service file /etc/systemd/system/k3s.service
[INFO]  systemd: Enabling k3s unit
Created symlink /etc/systemd/system/multi-user.target.wants/k3s.service → /etc/systemd/system/k3s.service.
[INFO]  systemd: Starting k3s
```

Verifying its status:

sudo systemctl status k3s

```
barbara@kube-master:~$ sudo systemctl status k3s
● k3s.service - Lightweight Kubernetes
     Loaded: loaded (/etc/systemd/system/k3s.service; enabled; vendor preset: enabled)
     Active: active (running) since Thu 2022-10-27 22:09:17 UTC; 1min 32s ago
       Docs: https://k3s.io
    Process: 1150 ExecStartPre=/bin/sh -xc ! /usr/bin/systemctl is-enabled --quiet nm-cloud-setup.service (code=exited, status=0/SUCCESS)
    Process: 1152 ExecStartPre=/sbin/modprobe br_netfilter (code=exited, status=0/SUCCESS)
    Process: 1156 ExecStartPre=/sbin/modprobe overlay (code=exited, status=0/SUCCESS)
   Main PID: 1157 (k3s-server)
      Tasks: 103
     Memory: 1.3G
        CPU: 27.682s
     CGroup: /system.slice/k3s.service
             ├─1157 "/usr/local/bin/k3s server"
             ├─1172 containerd -c /var/lib/rancher/k3s/agent/etc/containerd/config.toml -a /run/k3s/containerd/containerd.sock --state /run/k3s/containerd --root /var/lib/rancher/k3s/
agent/containerd
             ├─1866 /var/lib/rancher/k3s/data/2ef87ff954adbb390309ce4dc07500f29c319f84feec1719bfb5059c8808ec6a/bin/containerd-shim-runc-v2 -namespace k8s.io -id 0832234eda457ad714dc18
a849ae47f26687e1d1e5
             ├─1905 /var/lib/rancher/k3s/data/2ef87ff954adbb390309ce4dc07500f29c319f84feec1719bfb5059c8808ec6a/bin/containerd-shim-runc-v2 -namespace k8s.io -id b616dbf634e48ccb162aaa
bf15f76bfa7293dd01f5
             ├─1931 /var/lib/rancher/k3s/data/2ef87ff954adbb390309ce4dc07500f29c319f84feec1719bfb5059c8808ec6a/bin/containerd-shim-runc-v2 -namespace k8s.io -id 960e9f4bfda169491cd869
0ccc57036bc6b066c03e
             ├─2937 /var/lib/rancher/k3s/data/2ef87ff954adbb390309ce4dc07500f29c319f84feec1719bfb5059c8808ec6a/bin/containerd-shim-runc-v2 -namespace k8s.io -id c13bc081f8d8e211617939
6bd8f72e485f71c376a5
             └─3014 /var/lib/rancher/k3s/data/2ef87ff954adbb390309ce4dc07500f29c319f84feec1719bfb5059c8808ec6a/bin/containerd-shim-runc-v2 -namespace k8s.io -id 4c767f926b732f512ca70b
2844061e272922b69d65
```

Getting info of the nodes:

sudo kubectl get nodes

```
barbara@kube-master:~$ sudo kubectl get nodes
NAME           STATUS   ROLES                  AGE     VERSION
kube-master    Ready    control-plane,master   5m28s   v1.25.3+k3s1
barbara@kube-master:~$
```

Getting info of the namespaces

sudo kubectl get --all-namespaces

```
barbara@kube-master:~$ sudo kubectl get all --all-namespaces
NAMESPACE     NAME                                      READY   STATUS      RESTARTS   AGE
kube-system   pod/local-path-provisioner-5b5579c644-d4tfm   1/1     Running     0          27m
kube-system   pod/coredns-75fc8f8fff-dp5h8              1/1     Running     0          27m
kube-system   pod/helm-install-traefik-crd-krgh5        0/1     Completed   0          27m
kube-system   pod/svclb-traefik-b650ac6a-5jntr          2/2     Running     0          26m
kube-system   pod/helm-install-traefik-rb5d4            0/1     Completed   1          27m
kube-system   pod/metrics-server-5c8978b444-pvxs6       1/1     Running     0          27m
kube-system   pod/traefik-9c6dc6686-645sh               1/1     Running     0          26m

NAMESPACE     NAME                     TYPE           CLUSTER-IP      EXTERNAL-IP   PORT(S)                      AGE
default       service/kubernetes       ClusterIP      10.43.0.1       <none>        443/TCP                      27m
kube-system   service/kube-dns         ClusterIP      10.43.0.10      <none>        53/UDP,53/TCP,9153/TCP       27m
kube-system   service/metrics-server   ClusterIP      10.43.248.66    <none>        443/TCP                      27m
kube-system   service/traefik          LoadBalancer   10.43.33.208    10.0.2.5      80:30466/TCP,443:30790/TCP   26m

NAMESPACE     NAME                                DESIRED   CURRENT   READY   UP-TO-DATE   AVAILABLE   NODE SELECTOR   AGE
kube-system   daemonset.apps/svclb-traefik-b650ac6a   1         1         1       1            1           <none>          26m

NAMESPACE     NAME                                READY   UP-TO-DATE   AVAILABLE   AGE
kube-system   deployment.apps/local-path-provisioner   1/1     1            1           27m
kube-system   deployment.apps/coredns                  1/1     1            1           27m
kube-system   deployment.apps/metrics-server           1/1     1            1           27m
kube-system   deployment.apps/traefik                  1/1     1            1           26m

NAMESPACE     NAME                                           DESIRED   CURRENT   READY   AGE
kube-system   replicaset.apps/local-path-provisioner-5b5579c644   1         1         1       27m
kube-system   replicaset.apps/coredns-75fc8f8fff                 1         1         1       27m
kube-system   replicaset.apps/metrics-server-5c8978b444          1         1         1       27m
kube-system   replicaset.apps/traefik-9c6dc6686                  1         1         1       26m

NAMESPACE     NAME                              COMPLETIONS   DURATION   AGE
kube-system   job.batch/helm-install-traefik-crd   1/1           27s        27m
kube-system   job.batch/helm-install-traefik       1/1           31s        27m
```

Installing docker

Follow the commands of https://docs.docker.com/engine/install/ubuntu/
And
https://docs.docker.com/engine/install/linux-postinstall/

```
barbara@kube-master:~$ sudo apt-get install \
    ca-certificates \
    curl \
    gnupg \
    lsb-release
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
ca-certificates is already the newest version (20211016).
ca-certificates set to manually installed.
lsb-release is already the newest version (11.1.0ubuntu4).
lsb-release set to manually installed.
curl is already the newest version (7.81.0-1ubuntu1.6).
curl set to manually installed.
gnupg is already the newest version (2.2.27-3ubuntu2.1).
gnupg set to manually installed.
0 upgraded, 0 newly installed, 0 to remove and 45 not upgraded.
barbara@kube-master:~$ sudo mkdir -p /etc/apt/keyrings
barbara@kube-master:~$

curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o /etc/apt/keyrings/docker.gpg
barbara@kube-master:~$ echo \
  "deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.gpg] https://download.docker.com/linux/ubuntu \
  $(lsb_release -cs) stable" | sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
barbara@kube-master:~$ sudo apt-get update
Hit:1 http://br.archive.ubuntu.com/ubuntu jammy InRelease
Hit:2 http://br.archive.ubuntu.com/ubuntu jammy-updates InRelease
Hit:3 http://br.archive.ubuntu.com/ubuntu jammy-backports InRelease
Hit:4 http://br.archive.ubuntu.com/ubuntu jammy-security InRelease
Get:5 https://download.docker.com/linux/ubuntu jammy InRelease [48.9 kB]
Get:6 https://download.docker.com/linux/ubuntu jammy/stable amd64 Packages [9481 B]
Fetched 58.3 kB in 1s (64.0 kB/s)
Reading package lists... Done
```

```
barbara@kube-master:~$ sudo apt-get install docker-ce docker-ce-cli containerd.io docker-compose-plugin
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following packages were automatically installed and are no longer required:
  bridge-utils dns-root-data dnsmasq-base ubuntu-fan
Use 'sudo apt autoremove' to remove them.
The following additional packages will be installed:
  docker-ce-rootless-extras docker-scan-plugin libltdl7 libslirp0
  slirp4netns
Suggested packages:
  aufs-tools cgroupfs-mount | cgroup-lite
The following packages will be REMOVED:
  containerd docker.io runc
The following NEW packages will be installed:
  containerd.io docker-ce docker-ce-cli docker-ce-rootless-extras
  docker-compose-plugin docker-scan-plugin libltdl7 libslirp0
  slirp4netns
0 upgraded, 9 newly installed, 3 to remove and 45 not upgraded.
Need to get 111 MB of archives.
After this operation, 146 MB of additional disk space will be used.
Do you want to continue? [Y/n] Y
Get:1 http://br.archive.ubuntu.com/ubuntu jammy/main amd64 libltdl7 amd64 2.4.6-15build2 [39.6 kB]
Get:2 http://br.archive.ubuntu.com/ubuntu jammy/main amd64 libslirp0 amd64 4.6.1-1build1 [61.5 kB]
Get:3 http://br.archive.ubuntu.com/ubuntu jammy/universe amd64 slirp4netns amd64 1.0.1-2 [28.2 kB]
Get:4 https://download.docker.com/linux/ubuntu jammy/stable amd64 containerd.io amd64 1.6.9-1 [27.7 MB]
Get:5 https://download.docker.com/linux/ubuntu jammy/stable amd64 docker-ce-cli amd64 5:20.10.21~3-0~ubuntu-jammy [41.5 MB]
Get:6 https://download.docker.com/linux/ubuntu jammy/stable amd64 docker-ce amd64 5:20.10.21~3-0~ubuntu-jammy [20.5 MB]
Get:7 https://download.docker.com/linux/ubuntu jammy/stable amd64 docker-ce-rootless-extras amd64 5:20.10.21~3-0~ubuntu-jammy [8389
Get:8 https://download.docker.com/linux/ubuntu jammy/stable amd64 docker-compose-plugin amd64 2.12.2~ubuntu-jammy [9566 kB]
Get:9 https://download.docker.com/linux/ubuntu jammy/stable amd64 docker-scan-plugin amd64 0.21.0~ubuntu-jammy [3622 kB]
Fetched 111 MB in 34s (3304 kB/s)
(Reading database ... 73715 files and directories currently installed.)
```

```
Setting up libslirp0:amd64 (4.6.1-1build1) ...
Setting up docker-ce-rootless-extras (5:20.10.21~3-0~ubuntu-jammy) ...
Setting up slirp4netns (1.0.1-2) ...
Setting up docker-ce (5:20.10.21~3-0~ubuntu-jammy) ...
Job for docker.service failed because the control process exited with error code.
See "systemctl status docker.service" and "journalctl -xeu docker.service" for details.
invoke-rc.d: initscript docker, action "start" failed.
● docker.service - Docker Application Container Engine
     Loaded: loaded (/lib/systemd/system/docker.service; enabled; vendor preset: enabled)
     Active: activating (auto-restart) (Result: exit-code) since Thu 2022-10-27 22:43:33 UTC; 5ms ago
TriggeredBy: ● docker.socket
       Docs: https://docs.docker.com
    Process: 6241 ExecStart=/usr/bin/dockerd -H fd:// --containerd=/run/containerd/containerd.sock (code=exited, status=1/FAILURE)
   Main PID: 6241 (code=exited, status=1/FAILURE)
        CPU: 39ms
dpkg: error processing package docker-ce (--configure):
 installed docker-ce package post-installation script subprocess returned error exit status 1
Processing triggers for man-db (2.10.2-1) ...
Processing triggers for libc-bin (2.35-0ubuntu3.1) ...
Errors were encountered while processing:
 docker-ce
needrestart is being skipped since dpkg has failed
E: Sub-process /usr/bin/dpkg returned an error code (1)
barbara@kube-master:~$ sudo docker run hello-world
docker: Cannot connect to the Docker daemon at unix:///var/run/docker.sock. Is the docker daemon running?.
See 'docker run --help'.
```

Solution:

sudo groupadd docker
sudo usermod -aG docker $USER
newgrp docker

```
barbara@kube-master:~$ sudo groupadd docker
groupadd: group 'docker' already exists
barbara@kube-master:~$ sudo usermod -aG docker $USER
barbara@kube-master:~$ newgrp docker
```

```
barbara@kube-master:~$ sudo docker run hello-world
[sudo] password for barbara:
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
2db29710123e: Pull complete
Digest: sha256:e18f0a777aefabe047a671ab3ec3eed05414477c951ab1a6f352a06974245fe7
Status: Downloaded newer image for hello-world:latest

Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
 1. The Docker client contacted the Docker daemon.
 2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
    (amd64)
 3. The Docker daemon created a new container from that image which runs the
    executable that produces the output you are currently reading.
 4. The Docker daemon streamed that output to the Docker client, which sent it
    to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
 $ docker run -it ubuntu bash
```

Nice! :)

2. Install k3s in Worker nodes

Get the token from the Master node to build the cluster:

sudo cat /var/lib/rancher/k3s/server/node-token

```
barbara@kube-master:~$ sudo cat /var/lib/rancher/k3s/server/node-token
K102ff9c651f474accc3e96db99a370562f58b0e417500550c198feb8f7c0473a4a::server:64f07a9817abf8aaec07458a1404e694
barbara@kube-master:~$
```

Apply the command on the worker nodes, for the installation of k3s and ingress in the cluster:

curl -sfL https://get.k3s.io | K3S_URL=https://myserver:6443 K3S_TOKEN=mynodetoken sh -

worker1:

```
barbara@kube-worker1:~$ curl -sfL https://get.k3s.io | K3S_URL=https://192.168.3.20:6443 K3S_TOKEN=K102ff9c65
17abf8aaec07458a1404e694 sh -
[sudo] password for barbara:
[INFO]  Finding release for channel stable
[INFO]  Using v1.25.3+k3s1 as release
[INFO]  Downloading hash https://github.com/k3s-io/k3s/releases/download/v1.25.3+k3s1/sha256sum-amd64.txt
[INFO]  Downloading binary https://github.com/k3s-io/k3s/releases/download/v1.25.3+k3s1/k3s
[INFO]  Verifying binary download
[INFO]  Installing k3s to /usr/local/bin/k3s
[INFO]  Skipping installation of SELinux RPM
[INFO]  Creating /usr/local/bin/kubectl symlink to k3s
[INFO]  Creating /usr/local/bin/crictl symlink to k3s
[INFO]  Creating /usr/local/bin/ctr symlink to k3s
[INFO]  Creating killall script /usr/local/bin/k3s-killall.sh
[INFO]  Creating uninstall script /usr/local/bin/k3s-agent-uninstall.sh
[INFO]  env: Creating environment file /etc/systemd/system/k3s-agent.service.env
[INFO]  systemd: Creating service file /etc/systemd/system/k3s-agent.service
[INFO]  systemd: Enabling k3s-agent unit
Created symlink /etc/systemd/system/multi-user.target.wants/k3s-agent.service → /etc/systemd/system/k3s-agent
[INFO]  systemd: Starting k3s-agent
barbara@kube-worker1:~$
```

worker2:

```
barbara@kube-worker2:~$ curl -sfL https://get.k3s.io | K3S_URL=https://192.168.3.20:6443 K3S_TOKEN=K102ff9c651f474accc3e96
17abf8aaec07458a1404e694 sh -
[sudo] password for barbara:
[INFO]  Finding release for channel stable
[INFO]  Using v1.25.3+k3s1 as release
[INFO]  Downloading hash https://github.com/k3s-io/k3s/releases/download/v1.25.3+k3s1/sha256sum-amd64.txt
[INFO]  Downloading binary https://github.com/k3s-io/k3s/releases/download/v1.25.3+k3s1/k3s
[INFO]  Verifying binary download
[INFO]  Installing k3s to /usr/local/bin/k3s
[INFO]  Skipping installation of SELinux RPM
[INFO]  Creating /usr/local/bin/kubectl symlink to k3s
[INFO]  Creating /usr/local/bin/crictl symlink to k3s
[INFO]  Creating /usr/local/bin/ctr symlink to k3s
[INFO]  Creating killall script /usr/local/bin/k3s-killall.sh
[INFO]  Creating uninstall script /usr/local/bin/k3s-agent-uninstall.sh
[INFO]  env: Creating environment file /etc/systemd/system/k3s-agent.service.env
[INFO]  systemd: Creating service file /etc/systemd/system/k3s-agent.service
[INFO]  systemd: Enabling k3s-agent unit
Created symlink /etc/systemd/system/multi-user.target.wants/k3s-agent.service → /etc/systemd/system/k3s-agent.service.
[INFO]  systemd: Starting k3s-agent
barbara@kube-worker2:~$
```

Verifying the cluster creation in the master node and labeling them as worker nodes:

sudo kubectl get nodes
sudo kubectl label node kube-worker1 node-role.kubernetes.io/worker=worker

```
barbara@kube-master:~$ sudo kubectl get nodes
NAME           STATUS   ROLES                  AGE    VERSION
kube-worker1   Ready    <none>                 10m    v1.25.3+k3s1
kube-worker2   Ready    <none>                 2m5s   v1.25.3+k3s1
kube-master    Ready    control-plane,master   62m    v1.25.3+k3s1
```

```
barbara@kube-master:~$ sudo kubectl get nodes
NAME           STATUS   ROLES                  AGE    VERSION
kube-worker1   Ready    <none>                 10m    v1.25.3+k3s1
kube-worker2   Ready    <none>                 2m5s   v1.25.3+k3s1
kube-master    Ready    control-plane,master   62m    v1.25.3+k3s1
barbara@kube-master:~$ kubectl label node kube-worker1 node-role.kubernetes.io/worker=worker
WARN[0000] Unable to read /etc/rancher/k3s/k3s.yaml, please start server with --write-kubeconfig-mode to modify kube config permissions
error: error loading config file "/etc/rancher/k3s/k3s.yaml": open /etc/rancher/k3s/k3s.yaml: permission denied
barbara@kube-master:~$ sudo kubectl label node kube-worker1 node-role.kubernetes.io/worker=worker
node/kube-worker1 labeled
barbara@kube-master:~$ sudo kubectl label node kube-worker2 node-role.kubernetes.io/worker=worker
node/kube-worker2 labeled
barbara@kube-master:~$ sudo kubectl get nodes
NAME           STATUS   ROLES                  AGE     VERSION
kube-master    Ready    control-plane,master   63m     v1.25.3+k3s1
kube-worker1   Ready    worker                 11m     v1.25.3+k3s1
kube-worker2   Ready    worker                 3m32s   v1.25.3+k3s1
```

3. Install OpenFaaS

Using helm chart:

Helm is useful to install Kubernetes applications. It packages files to be installed in the helm charts.

arkade get helm

curl -sSLf https://raw.githubusercontent.com/helm/helm/master/scripts/get-helm-3 | bash

helm repo add openfaas https://openfaas.github.io/faas-netes/

helm repo update \
 && helm upgrade openfaas --install openfaas/openfaas \
    --namespace openfaas \
    --set functionNamespace=openfaas-fn \
    --set generateBasicAuth=true

```
root@kube-master:/home/barbara# helm upgrade openfaas --install openfaas/openfaas     --namespace openfaas     --set functionNamespace=openfaas-fn     --set generateBasicAuth=true
Release "openfaas" does not exist. Installing it now.
NAME: openfaas
LAST DEPLOYED: Fri Oct 28 00:15:57 2022
NAMESPACE: openfaas
STATUS: deployed
REVISION: 1
TEST SUITE: None
NOTES:
To verify that openfaas has started, run:

  kubectl -n openfaas get deployments -l "release=openfaas, app=openfaas"
To retrieve the admin password, run:

  echo $(kubectl -n openfaas get secret basic-auth -o jsonpath="{.data.basic-auth-password}" | base64 --decode)
root@kube-master:/home/barbara#
```

4. Install faas-cli

The faas-cli can be used to build and deploy functions to OpenFaaS:

curl sSL https://cli.openfaas.com | sudo sh

```
root@kube-master:~# curl -sSL https://cli.openfaas.com | sudo sh
Finding latest version from GitHub
0.14.11
Downloading package https://github.com/openfaas/faas-cli/releases/download/0.14.11/faas-cli as /tmp/faas-cli
Download complete.

Running with sufficient permissions to attempt to move faas-cli to /usr/local/bin
New version of faas-cli installed to /usr/local/bin
Creating alias 'faas' for 'faas-cli'.
  ___                   _____           ____
 / _ \ _ __   ___ _ __ |  ___|_ _  __ _/ ___|
| | | | '_ \ / _ \ '_ \| |_ / _` |/ _` \___ \
| |_| | |_) |  __/ | | |  _| (_| | (_| |___) |
 \___/| .__/ \___|_| |_|_|  \__,_|\__,_|____/
      |_|

CLI:
 commit:  8820d8e4a15dab900d8a7e8fc271851ccb94012e
 version: 0.14.11
```

Deployment of functions

0. Before everything:

Forward the port so you can connect to the gateway running inside our Kubernetes cluster:

kubectl rollout status -n openfaas deploy/gateway
kubectl port-forward -n openfaas svc/gateway 8080:8080 &

```
root@barbara-Inspiron-15-3511:/home/barbara# kubectl rollout status -n openfaas deploy/gateway
deployment "gateway" successfully rolled out
root@barbara-Inspiron-15-3511:/home/barbara# kubectl port-forward -n openfaas svc/gateway 8080:808
[1] 56062
root@barbara-Inspiron-15-3511:/home/barbara# Forwarding from 127.0.0.1:8080 -> 8080
Forwarding from [::1]:8080 -> 8080
Handling connection for 8080
Handling connection for 8080
Handling connection for 8080
Handling connection for 8080
Handling connection for 8080
Handling connection for 8080
```

Login to the faas-cli

PASSWORD=$(kubectl get secret -n openfaas basic-auth -o jsonpath="{.data.basic-auth-password}" | base64 --decode; echo)

env | grep PASSWORD

echo -n $PASSWORD | faas-cli login --username admin --password-stdin

```
root@kube-master:~/functions# PASSWORD=$(kubectl get secret -n openfaas basic-auth -o jsonpath="{.data.basic-auth-password}" | base64 --decode; echo)
root@kube-master:~/functions# env | grep PASSWORD
root@kube-master:~/functions# echo -n $PASSWORD | faas-cli login --username admin --password-stdin
Calling the OpenFaaS server to validate the credentials...
credentials saved for admin http://127.0.0.1:8080
```

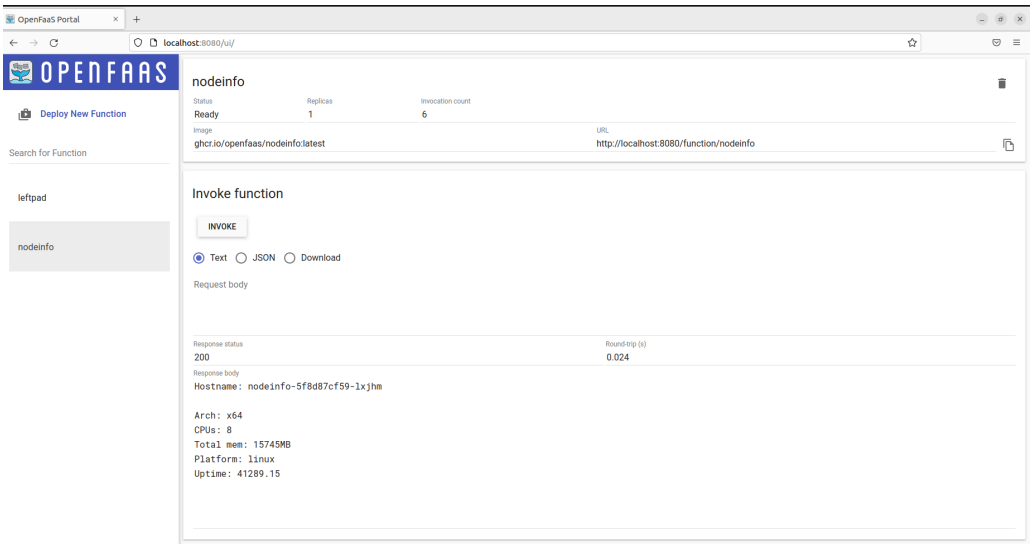
If you want, you can also use the UI.

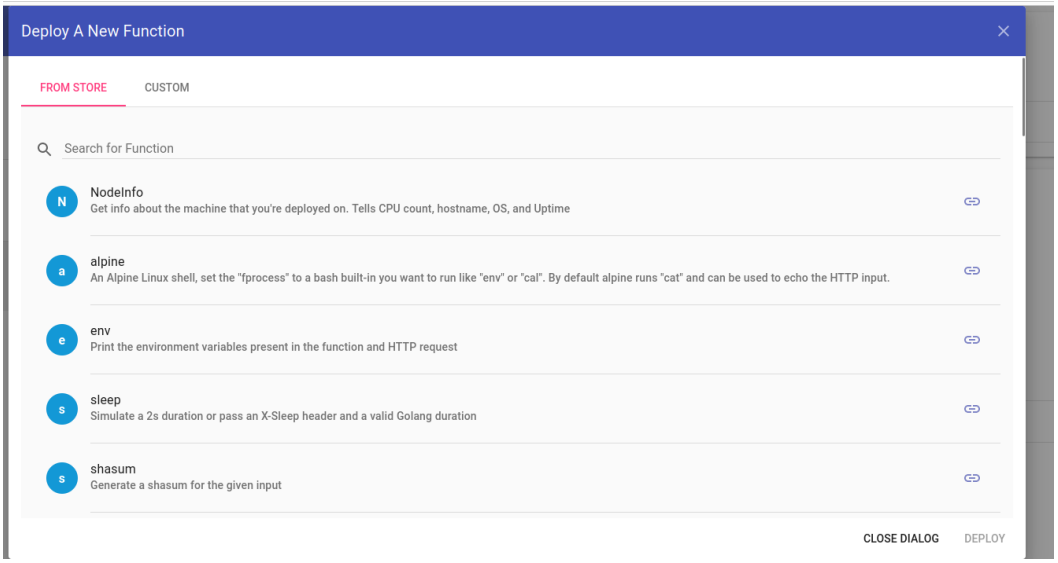
The login is admin. With this command, you get the passport to have access to the browser OpenFaaS UI:

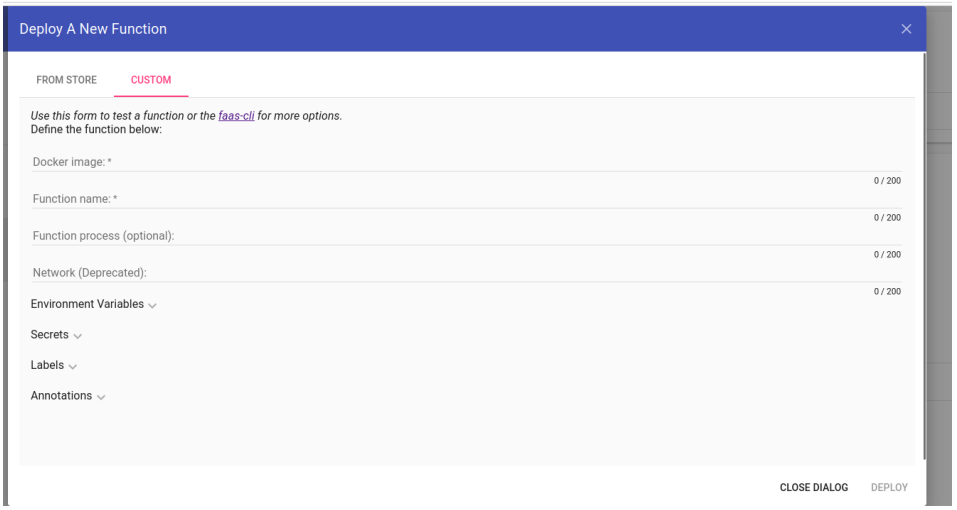
PASSWORD=$(kubectl get secret -n openfaas basic-auth -o jsonpath="{.data.basic-auth-password}" | base64 --decode; echo)echo -n $PASSWORD

In the browser, you can also deploy your functions:

http://localhost:8081/ui/







1. Installing the OpenFaaS MQTT-connector

MQTT-connector will trigger our function. It sub to a topic, and when there is a message via broker with this topic, the function is triggered. The command can install the mqtt-connector, specifying the topic, broker-host and client-id:

CLIENT_ID=$(head -c 12 /dev/urandom | shasum| cut -d' ' -f1)

arkade install mqtt-connector \
  --topics temperature-sensor \
  --broker-host tcp://test.mosquitto.org:1883 \
  --client-id $CLIENT_ID

For our test, we have a sensor publishing a message via MQTT, which we will be simulated by sending a temperature message as input.
Pull a template for python3-flask from faas-cli store:

faas-cli template store pull python3-flask

```
root@barbara-Inspiron-15-3511:/home/barbara# faas-cli template store pull python3-flask
Fetch templates from repository: https://github.com/openfaas/python-flask-template at
2022/10/29 19:50:56 Attempting to expand templates from https://github.com/openfaas/python-flas
-template
2022/10/29 19:50:57 Fetched 5 template(s) : [python27-flask python3-flask python3-flask-debian
ython3-http python3-http-debian] from https://github.com/openfaas/python-flask-template
root@barbara-Inspiron-15-3511:/home/barbara# faas-cli deploy
To deploy a function give --yaml/-f or a --image and --name flag
root@barbara-Inspiron-15-3511:/home/barbara# pip3 install paho-mqtt
Command 'pip3' not found, but can be installed with:
apt install python3-pip
root@barbara-Inspiron-15-3511:/home/barbara# apt install python3-pip
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following package was automatically installed and is no longer required:
  libnftables1
Use 'apt autoremove' to remove it.
The following additional packages will be installed:
```

Install paho-mqtt
Library we'll use for our function sensor.py

pip3 install paho-mqtt

```
root@kube-master:/home/barbara# pip3 install paho-mqtt
Collecting paho-mqtt
  Downloading paho-mqtt-1.6.1.tar.gz (99 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 99.4/99.4 KB 4.3 MB/s eta 0:00:00

  Preparing metadata (setup.py) ... done
Building wheels for collected packages: paho-mqtt
  Building wheel for paho-mqtt (setup.py) ... done
  Created wheel for paho-mqtt: filename=paho_mqtt-1.6.1-py3-none-any.whl size=62133
0a6
  Stored in directory: /root/.cache/pip/wheels/8b/bb/0c/79444d1dee20324d442856979b5b
Successfully built paho-mqtt
Installing collected packages: paho-mqtt
Successfully installed paho-mqtt-1.6.1
```

Sending message via MQTT, using our function sensor.py:

python3 sensor.py '{"sensor_id": 1, "temperature_c": 50}'

```
root@kube-master:/home/barbara# python3 sensor.py '{"sensor_id": 1, "temperature_c": 27}'
Connecting to test.mosquitto.org:1883
Message "{"sensor_id": 1, "temperature_c": 27}" published to "temperature-sensor"
root@kube-master:/home/barbara# python3 sensor.py '{"sensor_id": 1, "temperature_c": 23}'
Connecting to test.mosquitto.org:1883
Message "{"sensor_id": 1, "temperature_c": 23}" published to "temperature-sensor"
```

2. Build or deploy our function:

We can create, deploy a function available in faas-cli store.
In our example, the function will print the message we receive via MQTT:

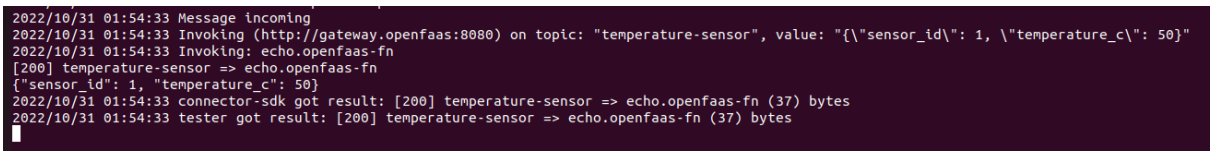
faas-cli deploy --name echo --image ghcr.io/openfaas/alpine:latest \
  --fprocess=cat \
  --annotation topic="temperature-sensor"

```
root@kube-master:/home/barbara# faas-cli deploy --name echo --image ghcr.io/openfaas/alpine:latest \
  --fprocess=cat \
  --annotation topic="temperature-sensor"

Deployed. 202 Accepted.
URL: http://127.0.0.1:8080/function/echo
```

Getting the logs of our deployment
kubectl logs deploy/mqtt-connector -n openfaas -f

```
root@kube-master:/home/barbara# kubectl logs deploy/mqtt-connector -n openfaas -f
2022/10/30 18:20:05 Topic: "temperature-sensor" Broker: "tcp://test.mosquitto.org:1883"
2022/10/30 18:20:05 Gateway: http://gateway.openfaas:8080       Async: false
2022/10/30 18:20:08 Connection requested for broker: tcp://test.mosquitto.org:1883
2022/10/30 18:20:08 Connected to tcp://test.mosquitto.org:1883
2022/10/30 18:20:08 Subscribed to topic: temperature-sensor
2022/10/30 18:20:22 Message incoming
2022/10/30 18:20:22 Invoking (http://gateway.openfaas:8080) on topic: "temperature-sensor", value: "{\"sensor_id\": 1, \"temperature_c\": 27}"
2022/10/30 18:20:52 Message incoming
2022/10/30 18:20:52 Invoking (http://gateway.openfaas:8080) on topic: "temperature-sensor", value: "{\"sensor_id\": 1, \"temperature_c\": 23}"
```


2022/10/31 01:54:33 Message incoming
2022/10/31 01:54:33 Invoking (http://gateway.openfaas:8080) on topic: "temperature-sensor", value: "{\"sensor_id\": 1, \"temperature_c\": 50}"
2022/10/31 01:54:33 Invoking: echo.openfaas-fn
[200] temperature-sensor => echo.openfaas-fn
{"sensor_id": 1, "temperature_c": 50}
2022/10/31 01:54:33 connector-sdk got result: [200] temperature-sensor => echo.openfaas-fn (37) bytes
2022/10/31 01:54:33 tester got result: [200] temperature-sensor => echo.openfaas-fn (37) bytes