MongoDB

Ahhh Mongo. First they think you're crazy, then they fight you and then you change the world. MongoDB is a NoSQL Database system that is flexible, can work virtually everywhere and is tough.

The deployment is pretty easy to be honest, so just relax and take it in.

Let's start!

Overview and Requirements

We will have to create a new namespace "mongodb" and deploy the bitnami version of MongoDb

You need to have already red the Local StorageClass situation. If not, please read this guide (Local Storage (any why you are getting the error 0 /1 deployed)) to familiarize yourself with the constraints of our local K8s cluster.

Before starting, please create the namespace mongodb by using the command **kubectl create namespace mongodb**

Deploy mongoDB

First of all, we have to edit the values.yaml file in order to configure the mongoDB deployment to our k8s environment.

For the ready made file, please check our gitlab (https://gitlab.f-in.io/finot/kubernetes/-/blob/master/mongodb/values.yaml)

If you need to create the values.yaml file yourself, download the values.yaml example file from the bitnami chart repository by using `wget https://github.com/bitnami/charts/raw/master/bitnami/mongodb/values.yaml`

⚠️ IMPORTANT INFO⚠️

We cannot use the latest version of MongoDB in our LocalK8s Cluster because it need Intel AVX capabilities, something we don't really have. We are going to use **version 4.4** as it is the last version that does not need these kinds of capabilities. Dont mess with latest. YOU HAVE BEEN INFORMED.

In order to change the version, please change the image tag in the values.yaml file to `4.4.15-debian-10-r8`

Then, in auth, change the rootUser and rootPassword of the example file.

Finaly, in persistance, change the mountPath to `/home/fint/storage/mongo`

Before deploying, please create the folder mongo in /home/fint/storage in the host machine.

Then deploy with `helm install mongo --namespace mongodb bitnami/mongodb -f ./values.yaml`

The output should be:

```
NAME: mongo
LAST DEPLOYED: Mon Aug  8 18:11:08 2022
NAMESPACE: mongodb
STATUS: deployed
REVISION: 1
TEST SUITE: None
NOTES:
CHART NAME: mongodb
CHART VERSION: 12.1.26
APP VERSION: 5.0.9

** Please be patient while the chart is being deployed **

MongoDB&reg; can be accessed on the following DNS name(s) and ports
from within your cluster:

    mongo-mongodb.mongodb.svc.cluster.local

To get the root password run:

    export MONGODB_ROOT_PASSWORD=$(kubectl get secret --namespace
mongodb mongo-mongodb -o jsonpath="{.data.mongodb-root-password}" |
base64 -d)

To connect to your database, create a MongoDB&reg; client container:

    kubectl run --namespace mongodb mongo-mongodb-client --rm --tty -i
--restart='Never' --env="MONGODB_ROOT_PASSWORD=$MONGODB_ROOT_PASSWORD"
--image docker.io/bitnami/mongodb:4.4.15-debian-10-r8 --command -- bash

Then, run the following command:
    mongosh admin --host "mongo-mongodb" --authenticationDatabase admin
-u root -p $MONGODB_ROOT_PASSWORD

To connect to your database from outside the cluster execute the
following commands:

    kubectl port-forward --namespace mongodb svc/mongo-mongodb 27017:
27017 &
    mongosh --host 127.0.0.1 --authenticationDatabase admin -p
$MONGODB_ROOT_PASSWORD
```

In some minutes the pods will be created, but the will not run. This, is actually normal at this stage. As said in the Local StorageClass guide (Local Storage (any why you are getting the error 0/1 deployed)) we have to create a PersistentVolume for the PVC that the helm chart has created.

mongo-pv.yaml

```
apiVersion: v1
kind: PersistentVolume
metadata:
  name: mongodb
  labels:
    app: mongodb
    type: local
  finalizers:
    - kubernetes.io/pv-protection
spec:
  capacity:
    storage: 8Gi
  hostPath:
    path: /home/fint/storage/mongodb
    type: ''
  accessModes:
    - ReadWriteOnce
  persistentVolumeReclaimPolicy: Retain
  storageClassName: standard
  volumeMode: Filesystem
```

With the above, we are creating a PersistentVolume named mongodb that has 8Gb storage capacity in the path /home/fint/storage/mongodb, with no specific type, that Reads and Writes sequentially, that will not be deleted if the volume is reclaimed by another PVC and uses the Storage Class “standard”. Now let’s apply our PersistentVolume by using **kubectl apply -f**

After some minutes, the MongoDB Pods should be up and running, but are we sure that MongoDB actually works?

Testing our MongoDB deployment

First of all, we need to export our MongoDB root password to the terminal. I know that you know the password, but you might want to test another MongoDB deployment that you do not remember the password, so bare with me.

Use the command `export MONGODB_ROOT_PASSWORD=$(kubectl get secret --namespace mongodb mongo-mongodb -o jsonpath="{.data.mongodb-root-password}" | base64 -d)`

and then run `kubectl run --namespace mongodb mongo-mongodb-client --rm --tty -i --restart='Never' --env="MONGODB_ROOT_PASSWORD=$MONGODB_ROOT_PASSWORD" --image docker.io/bitnami/mongodb:4.4.15-debian-10-r8 --command -- bash`

the output should be something like this:

```
If you don't see a command prompt, try pressing enter.
I have no name!@mongo-mongodb-client:/$ mongosh admin --host "mongo-
mongodb" --authenticationDatabase admin -u root -p
$MONGODB_ROOT_PASSWORD
Current Mongosh Log ID: 62f128c98c4f6b5619d2f74e
Connecting to:          mongodb://<credentials>@mongo-mongodb:27017
/admin?directConnection=true&authSource=admin&appName=mongosh+1.5.0
Using MongoDB:          4.4.15
Using Mongosh:          1.5.0

For mongosh info see: https://docs.mongodb.com/mongodb-shell/


To help improve our products, anonymous usage data is collected and
sent to MongoDB periodically (https://www.mongodb.com/legal/privacy-
policy).
You can opt-out by running the disableTelemetry() command.

------
   Enable MongoDB's free cloud-based monitoring service, which will
then receive and display
   metrics about your deployment (disk utilization, CPU, operation
statistics, etc).

   The monitoring data will be available on a MongoDB website with a
unique URL accessible to you
   and anyone you share the URL with. MongoDB may use this information
to make product
   improvements and to suggest MongoDB products and deployment options
to you.

   To enable free monitoring, run the following command: db.
enableFreeMonitoring()
   To permanently disable this reminder, run the following command: db.
disableFreeMonitoring()
------
```

From the output we can understand that mongoose has established a connection with our database. You can try imputing some MongoDB command, though the successful connection is a pretty good indication that MongoDB works.