Demo Project: Deploy MongoDB and Mongo Express into local K8s cluster

This guide demonstrates how to deploy MongoDB and MongoExpress on a Minikube Kubernetes cluster, complete with secure configurations using ConfigMaps and Secrets.

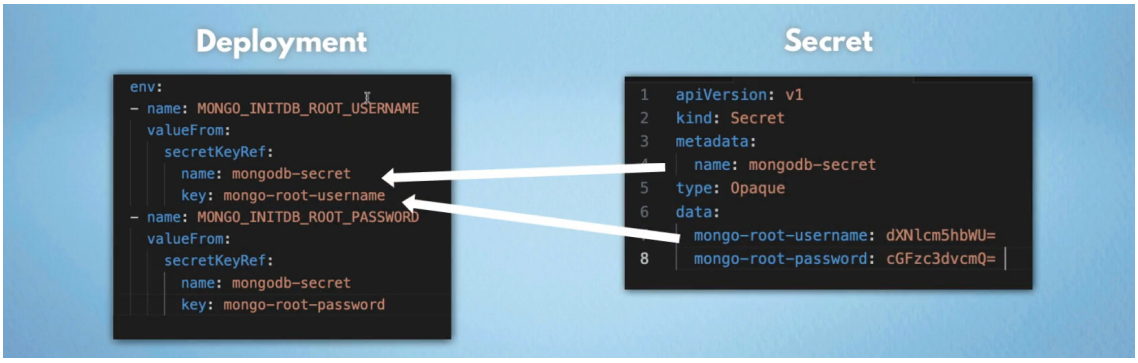
Step 1: Installation & create Minikube cluster

1. **Install Minikube**:
    - Follow the Minikube installation guide (*https://minikube.sigs.k8s.io/docs/start/*) for your operating system.
2. **Use Docker as the Driver**
    - Ensure Docker is installed on your system (preferred method). For more information, refer to Minikube drivers documentation (https://minikube.sigs.k8s.io/docs/drivers/)
3. **Start the Minikube Cluster**: `minikube start --driver docker`
4. **Verify Cluster Status**: `minikube status`

> Note: The kubectl CLI tool is installed as part of Minikube. You'll use kubectl for interacting with the cluster.

Step 2: Deploy MongoDB Using Secrets

1. **Verify the Minikube Cluster:** `kubectl get all`
2. **Check the specifications of the configuration and port for mongo-db.**
   - Go to Docker Hub (https://hub.docker.com/_/mongo)
   - Port is `27017`
   - **Environment Variables to create a new user and set that user's password:** `MONGO_INITDB_ROOT_USERNAME`, `MONGO_INITDB_ROOT_PASSWORD`
3. **Create a Secret for MongoDB**:
   - Encode credentials using base64:
     - `echo -n 'username' | base64`
     - `echo -n 'password' | base64`
   - cd into the directory and update `mongo-secret.yaml` with the encoded values.
   - Apply the Secret: `kubectl apply -f mongo-secret.yaml`
   - Confirm the Secret was created: `kubectl get secret`
4. **Deploy MongoDB**:
   - Update `mongo.yaml` to reference the Secrets.



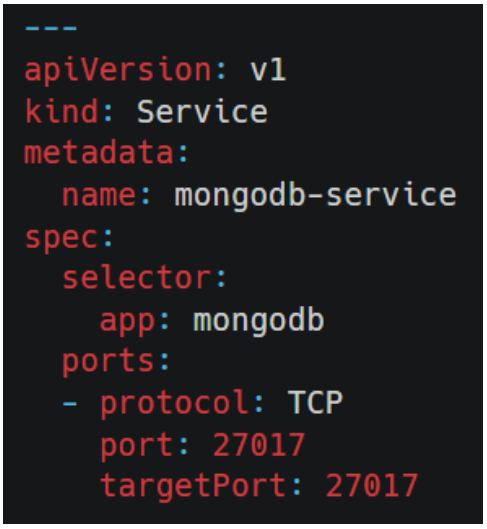
   - **Apply the Deployment:** `kubectl apply -f mongo.yaml`
   - **Verify Deployment:**

- `kubectl get all`
- `kubectl get pod`
- Get in more detail about a specific pod: `kubectl describe pod <pod name>`

Step 3: Create an internal service for MongoDB

Used for other components or other pods can talk to.

1. **Define an Internal Service**:
   - Add the following configuration to `mongo.yaml`:



```yaml
---
apiVersion: v1
kind: Service
metadata:
  name: mongodb-service
spec:
  selector:
    app: mongodb
  ports:
  - protocol: TCP
    port: 27017
    targetPort: 27017
```

2. **Apply the Service**: `kubectl apply -f mongo.yaml`
3. **Verify the Service**:
   - `kubectl get service`
   - Validate that the service is attached to the correct pod:
     `kubectl describe service <service name>`

     Example: `kubectl describe service mongodb-service`
4. **Validate the IP Address:** `kubectl get pod -o wide`

Step 4: Deploy MongoExpress with ConfigMap

1. **Check the specifications of the configuration and port for MongoDB.**
    - Go to Docker Hub (https://hub.docker.com/_/mongo-express)
    - Port is `8081`
    - For MongoDB address: `ME_CONFIG_MONGODB_SERVER`
    - Credentials to authenticate to the MongoDB:

      `ME_CONFIG_MONGODB_ADMINUSERNAME`

      `ME_CONFIG_MONGODB_ADMINPASSWORD`
2. **Create a ConfigMap for MongoExpress:**
    - Update `mongo-configmap.yaml` with the MongoDB service name and credentials.
    - Apply the ConfigMap: `kubectl apply -f mongo-configmap.yaml`
3. **Deploy MongoExpress**:
    - Define the deployment in `mongo-express.yaml`, referencing the ConfigMap. Also add the port and the 3 environment variables that mongo-express needs to connect and authenticate to the MongoDB.
    - Apply the Deployment: `kubectl apply -f mongo-express.yaml`
4. **Verify Deployment**:
    - `kubectl get pod`
    - Check logs: `kubectl logs <pod name>` to confirm that the Mongo Express server started.

Step 5: Expose MongoExpress via an External Service

1. **Add External Service Configuration**
    - Update `mongo-empress.yaml`

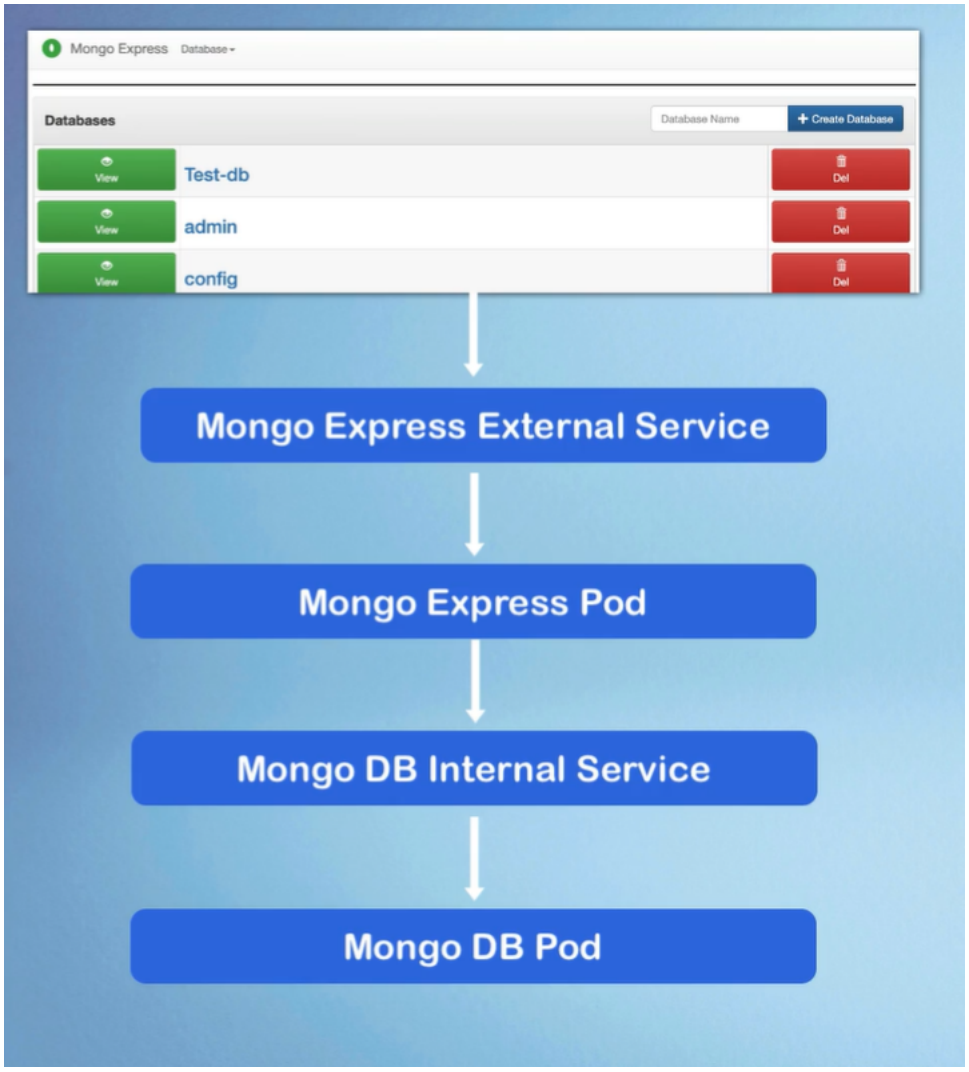
      Note:

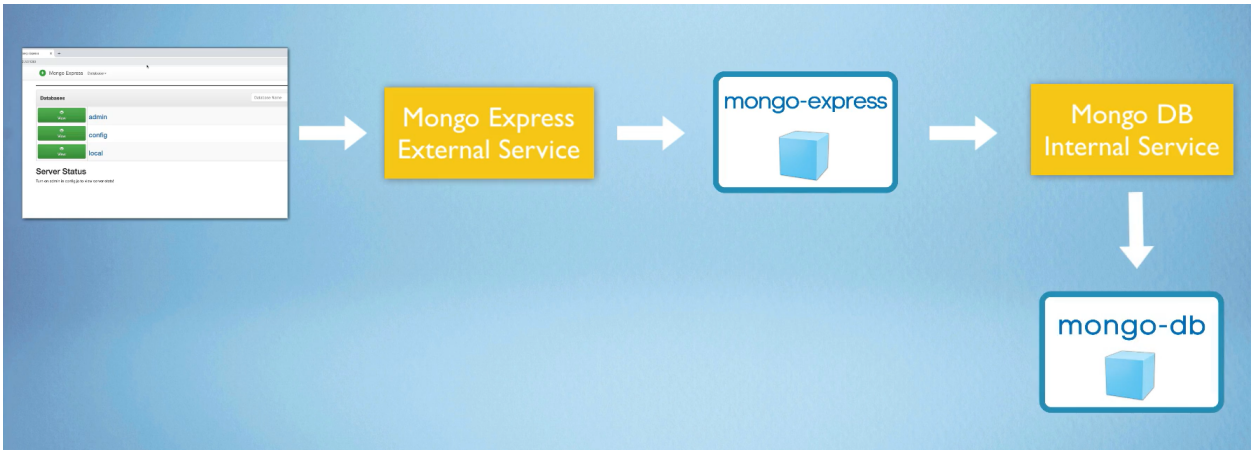
        - Setting `type: LoadBalancer` accepts external requests and assigns the service an external IP address.
            - However, Minikube does not assign an actual external IP for `LoadBalancer` services like a cloud-based Kubernetes setup does. Instead, the `minikube service` command assigns a local address.
        - Adding a `nodePort` within the range `30000-32767` is Kubernetes' reserved range for exposing services on specific ports via NodePort.
2. **Apply the Service Configuration**: `kubectl apply -f mongo-express.yaml`
3. **Open the Service in a Browser**: `minikube service mongo-express-service`

   Note: Minikube runs as a local Kubernetes cluster, and it does not automatically assign external IPs to `LoadBalancer` services. Running this command opens the service in a browser and assigns a temporary public IP or URL for accessing the service externally in your local environment.
4. **Log in to MongoExpress:**
    - Username: `admin`
    - Password: `pass`

Step 6: Testing the Setup

1. **Add a new database called:** `Test-db`.
2. **Deployment Flow**:
   Adding a database in MongoExpress triggers updates to the MongoDB deployment, creating this flow:


Mongo Express
Database
Databases
Database Name
Create Database
View
Test-db
Del
admin
config
Mongo Express External Service
Mongo Express Pod
Mongo DB Internal Service
Mongo DB Pod


Mongo Express External Service
mongo-express
Mongo DB Internal Service
mongo-db