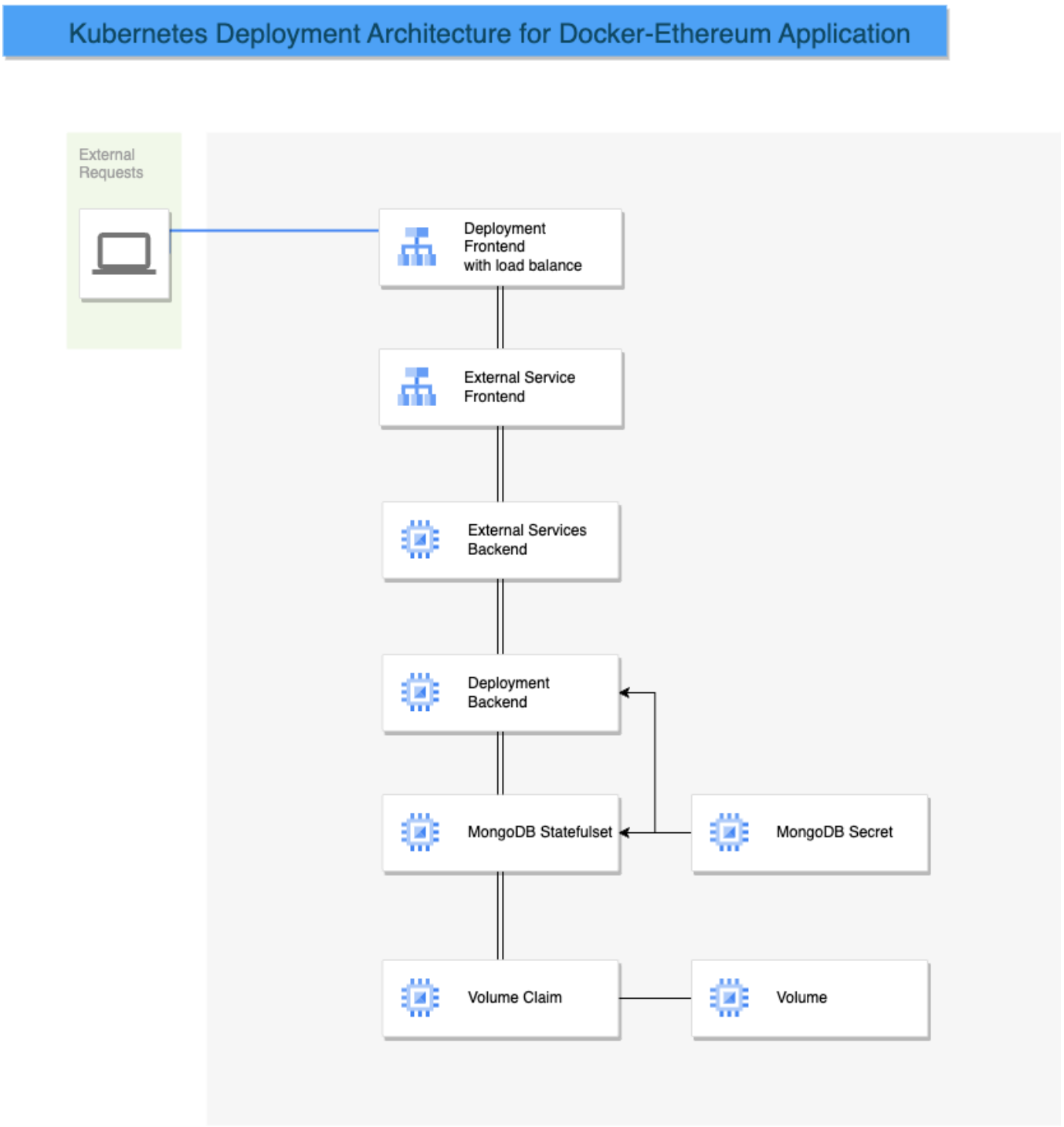
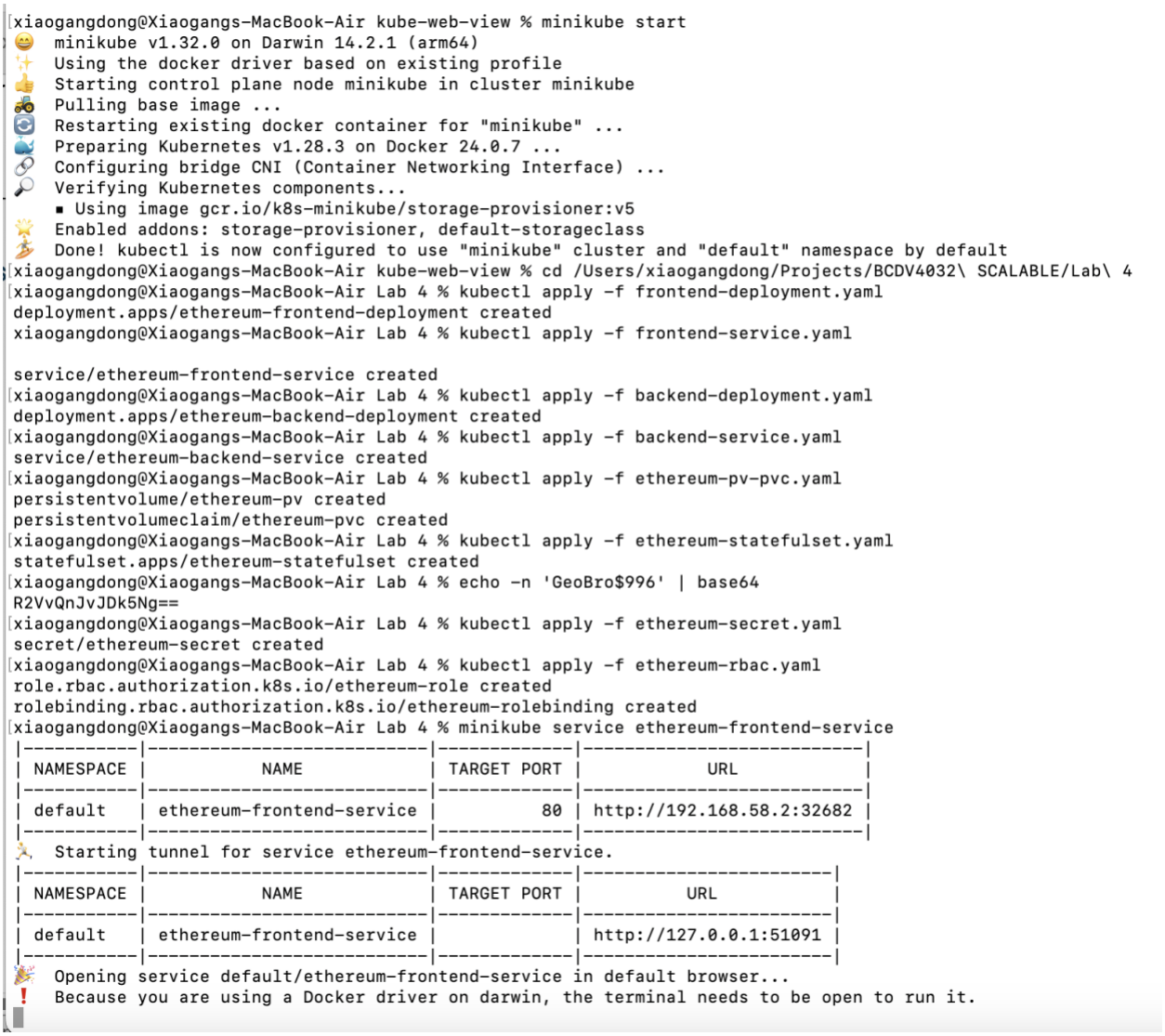
Kubernetes Deployment Architecture for Docker-Ethereum Application

By Xiaogang Dong (101492108)



Item	Design Rational
Deployment Option	StatefulSets: My ethereum app would need storage for the blockchain data. ReplicaSets are for stateless client nodes and don't serve the purpose.
Storage	PVC: use PVC together with Statefulsets to ensure each node has dedicated storage this is consistent. When a PVC created, it will be satisfied by existing PV which is at cluster level.
Scaling	Horizontal Pod Autoscaler: to ensure the app can scale in and out based on CPU or memory usage. It should be configured to scale automatically.
Load Balancing	Distribute the incoming requests evenly across the nodes to ensure the resources are utlized effectiently. The Kubernetes services should be defined as LoadBalancer type then they will be exposed to external
Secrets	The secret data is for storage of private keys or API credentials. Those types of data is sensitive. They should be mounted into pods as environment variables or files to ensure confidentiality.
User Roles and Access Control	Ensure the access to Kubernetes resources is secure and controllable. Levels of access should be set to fulfill the requirement of operational management (Read, write, Execute). The roles should be created then bind with respective service accounts using RoleBindings or ClusterRoleBindings.

Screenshots implementation of the design:



```
xiaogangdong@Xiaogangs-MacBook-Air kube-web-view % minikube start
😄  minikube v1.32.0 on Darwin 14.2.1 (arm64)
✨  Using the docker driver based on existing profile
👍  Starting control plane node minikube in cluster minikube
🚜  Pulling base image ...
🔄  Restarting existing docker container for "minikube" ...
🐳  Preparing Kubernetes v1.28.3 on Docker 24.0.7 ...
🔗  Configuring bridge CNI (Container Networking Interface) ...
🔎  Verifying Kubernetes components...
    ▪ Using image gcr.io/k8s-minikube/storage-provisioner:v5
🌟  Enabled addons: storage-provisioner, default-storageclass
🏄  Done! kubectl is now configured to use "minikube" cluster and "default" namespace by default
xiaogangdong@Xiaogangs-MacBook-Air kube-web-view % cd /Users/xiaogangdong/Projects/BCDV4032\ SCALABLE/Lab\ 4
xiaogangdong@Xiaogangs-MacBook-Air Lab 4 % kubectl apply -f frontend-deployment.yaml
deployment.apps/ethereum-frontend-deployment created
xiaogangdong@Xiaogangs-MacBook-Air Lab 4 % kubectl apply -f frontend-service.yaml

service/ethereum-frontend-service created
xiaogangdong@Xiaogangs-MacBook-Air Lab 4 % kubectl apply -f backend-deployment.yaml
deployment.apps/ethereum-backend-deployment created
xiaogangdong@Xiaogangs-MacBook-Air Lab 4 % kubectl apply -f backend-service.yaml
service/ethereum-backend-service created
xiaogangdong@Xiaogangs-MacBook-Air Lab 4 % kubectl apply -f ethereum-pv-pvc.yaml
persistentvolume/ethereum-pv created
persistentvolumeclaim/ethereum-pvc created
xiaogangdong@Xiaogangs-MacBook-Air Lab 4 % kubectl apply -f ethereum-statefulset.yaml
statefulset.apps/ethereum-statefulset created
xiaogangdong@Xiaogangs-MacBook-Air Lab 4 % echo -n 'GeoBro$996' | base64
R2VvQnJvJDk5Ng==
xiaogangdong@Xiaogangs-MacBook-Air Lab 4 % kubectl apply -f ethereum-secret.yaml
secret/ethereum-secret created
xiaogangdong@Xiaogangs-MacBook-Air Lab 4 % kubectl apply -f ethereum-rbac.yaml
role.rbac.authorization.k8s.io/ethereum-role created
rolebinding.rbac.authorization.k8s.io/ethereum-rolebinding created
xiaogangdong@Xiaogangs-MacBook-Air Lab 4 % minikube service ethereum-frontend-service
|-----------|---------------------------|-------------|---------------------------|
| NAMESPACE |           NAME            | TARGET PORT |            URL            |
|-----------|---------------------------|-------------|---------------------------|
| default   | ethereum-frontend-service |          80 | http://192.168.58.2:32682 |
|-----------|---------------------------|-------------|---------------------------|
🏃  Starting tunnel for service ethereum-frontend-service.
|-----------|---------------------------|-------------|------------------------|
| NAMESPACE |           NAME            | TARGET PORT |          URL           |
|-----------|---------------------------|-------------|------------------------|
| default   | ethereum-frontend-service |             | http://127.0.0.1:51091 |
|-----------|---------------------------|-------------|------------------------|
🎉  Opening service default/ethereum-frontend-service in default browser...
❗  Because you are using a Docker driver on darwin, the terminal needs to be open to run it.
```

```
xiaogangdong@Xiaogangs-MacBook-Air kube-web-view % cd /Users/xiaogangdong/Projects/BCDV4032\ SCALABLE/Lab\ 4
xiaogangdong@Xiaogangs-MacBook-Air Lab 4 % kubectl get svc
NAME                        TYPE           CLUSTER-IP      EXTERNAL-IP   PORT(S)          AGE
ethereum-backend-service    ClusterIP      10.96.149.90    <none>        80/TCP           12m
ethereum-frontend-service   LoadBalancer   10.104.60.172   <pending>     80:32682/TCP     14m
kubernetes                  ClusterIP      10.96.0.1       <none>        443/TCP          6d7h
mongodb-service             ClusterIP      10.110.83.89    <none>        27017/TCP        6d7h
note-server-service         ClusterIP      10.110.101.56   <none>        5000/TCP         6d7h
note-service                LoadBalancer   10.110.13.20    <pending>     3000:30444/TCP   6d7h
xiaogangdong@Xiaogangs-MacBook-Air Lab 4 % kubectl get deployment
NAME                         READY   UP-TO-DATE   AVAILABLE   AGE
ethereum-backend-deployment  2/2     2            2           13m
ethereum-frontend-deployment 2/2     2            2           16m
note-deployment              2/2     2            2           6d7h
note-server-deployment       2/2     2            2           6d7h
xiaogangdong@Xiaogangs-MacBook-Air Lab 4 % kubectl get statefulsets
NAME                  READY   AGE
ethereum-statefulset  1/1     11m
mongodb-stateful-set  2/2     6d7h
xiaogangdong@Xiaogangs-MacBook-Air Lab 4 % kubectl get pvc
NAME                                      STATUS   VOLUME                                     CAPACITY   ACCESS MODES   STORAGECLASS   AGE
ethereum-pvc                              Bound    pvc-23d0eafb-799f-40da-82b8-4f9084774cd3   1Gi        RWO            standard       12m
ethereum-storage-ethereum-statefulset-0   Bound    pvc-6e9eb6ce-48fb-4184-865c-b68c9bf9af33   1Gi        RWO            standard       12m
mongodb-data-mongodb-stateful-set-0       Bound    pvc-0bb6ffaa-aadb-47b7-ac71-c4be1450e930   1Gi        RWO            standard       6d7h
mongodb-data-mongodb-stateful-set-1       Bound    pvc-87a4f91e-6735-4783-8ad3-b00d7a3b17b7   1Gi        RWO            standard       4d1h
xiaogangdong@Xiaogangs-MacBook-Air Lab 4 % minikube ip
192.168.58.2
xiaogangdong@Xiaogangs-MacBook-Air Lab 4 %
```

YAML:

frontend-deployment.yaml × | frontend-service.yaml

frontend-deployment.yaml

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: ethereum-frontend-deployment
spec:
  replicas: 2
  selector:
    matchLabels:
      app: ethereum-frontend
  template:
    metadata:
      labels:
        app: ethereum-frontend
    spec:
      containers:
      - name: ethereum-frontend
        image: 0xdong/myfirstapp
        ports:
        - containerPort: 80
```

frontend-service.yaml

```yaml
apiVersion: v1
kind: Service
metadata:
  name: ethereum-frontend-service
spec:
  type: LoadBalancer
  selector:
    app: ethereum-frontend
  ports:
  - protocol: TCP
    port: 80
    targetPort: 80
```

backend-deployment.yaml

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: ethereum-backend-deployment
spec:
  replicas: 2
  selector:
    matchLabels:
      app: ethereum-backend
  template:
    metadata:
      labels:
        app: ethereum-backend
    spec:
      containers:
      - name: ethereum-backend
        image: 0xdong/myfirstapp
        ports:
        - containerPort: 80

```

backend-service.yaml

```yaml
apiVersion: v1
kind: Service
metadata:
  name: ethereum-backend-service
spec:
  selector:
    app: ethereum-backend
  ports:
  - protocol: TCP
    port: 80
    targetPort: 80

```

ethereum-pv-pvc.yaml

```yaml
apiVersion: v1
kind: PersistentVolume
metadata:
  name: ethereum-pv
spec:
  capacity:
    storage: 1Gi
  accessModes:
    - ReadWriteOnce
  hostPath:
    path: /mnt/data

---
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: ethereum-pvc
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 1Gi

```

ethereum-statefulset.yaml

```yaml
apiVersion: apps/v1
kind: StatefulSet
metadata:
  name: ethereum-statefulset
spec:
  serviceName: "ethereum-backend"
  replicas: 1
  selector:
    matchLabels:
      app: ethereum-backend
  template:
    metadata:
      labels:
        app: ethereum-backend
    spec:
      containers:
      - name: ethereum
        image: 0xdong/myfirstapp
        ports:
        - containerPort: 30303
        volumeMounts:
        - name: ethereum-storage
          mountPath: /var/ethereum
  volumeClaimTemplates:
  - metadata:
      name: ethereum-storage
    spec:
      accessModes: [ "ReadWriteOnce" ]
      resources:
        requests:
          storage: 1Gi
```

ethereum-secret.yaml

```yaml
apiVersion: v1
kind: Secret
metadata:
  name: ethereum-secret
type: Opaque
data:
  secretKey: R2VvQnJvJDk5Ng==

```

ethereum-rbac.yaml

```yaml
apiVersion: rbac.authorization.k8s.io/v1
kind: Role
metadata:
  namespace: default
  name: ethereum-role
rules:
- apiGroups: [""]
  resources: ["pods", "services"]
  verbs: ["get", "list", "watch", "create", "delete"]

---
apiVersion: rbac.authorization.k8s.io/v1
kind: RoleBinding
metadata:
  name: ethereum-rolebinding
  namespace: default
subjects:
- kind: ServiceAccount
  name: default
  namespace: default
roleRef:
  kind: Role
  name: ethereum-role
  apiGroup: rbac.authorization.k8s.io

```