Kubernetes有状态集群服务部署与管理

时速云：张寿红

Agenda

- Background
  - What is Kubernetes?
  - Run stateful services on Kubernetes
- Kubernetes Stateful Service Features
  - Init Container
  - Pet Set
- Kubernetes Storage
  - Volume
  - Persistent Volume
  - Dynamic Storage Provision
- Run MySQL Cluster on Kubernetes
  - Galera MySQL Introduction
  - Deploy MySQL Cluster with PetSet
  - Cluster Operation on Kubernetes

What is Kubernetes

Greek for *“Helmsman”*; also the root of the words *“governor”* and *“cybernetic”*

Runs and manages containers

Handles failure

Inspired and informed by Google’s experiences and internal systems

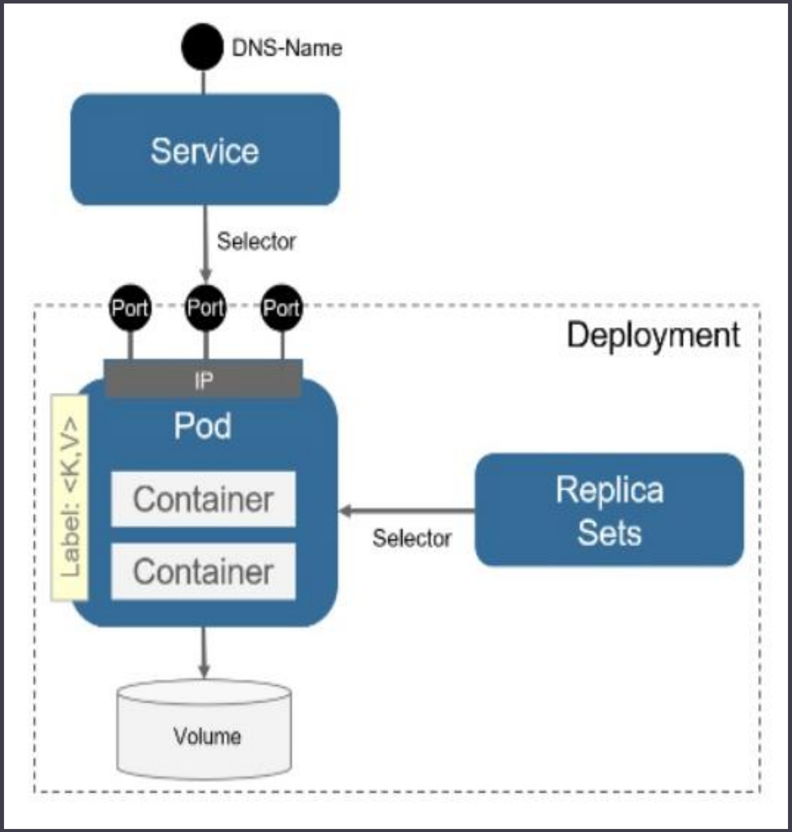
Supports multiple cloud and bare-metal environments

Supports multiple container runtimes

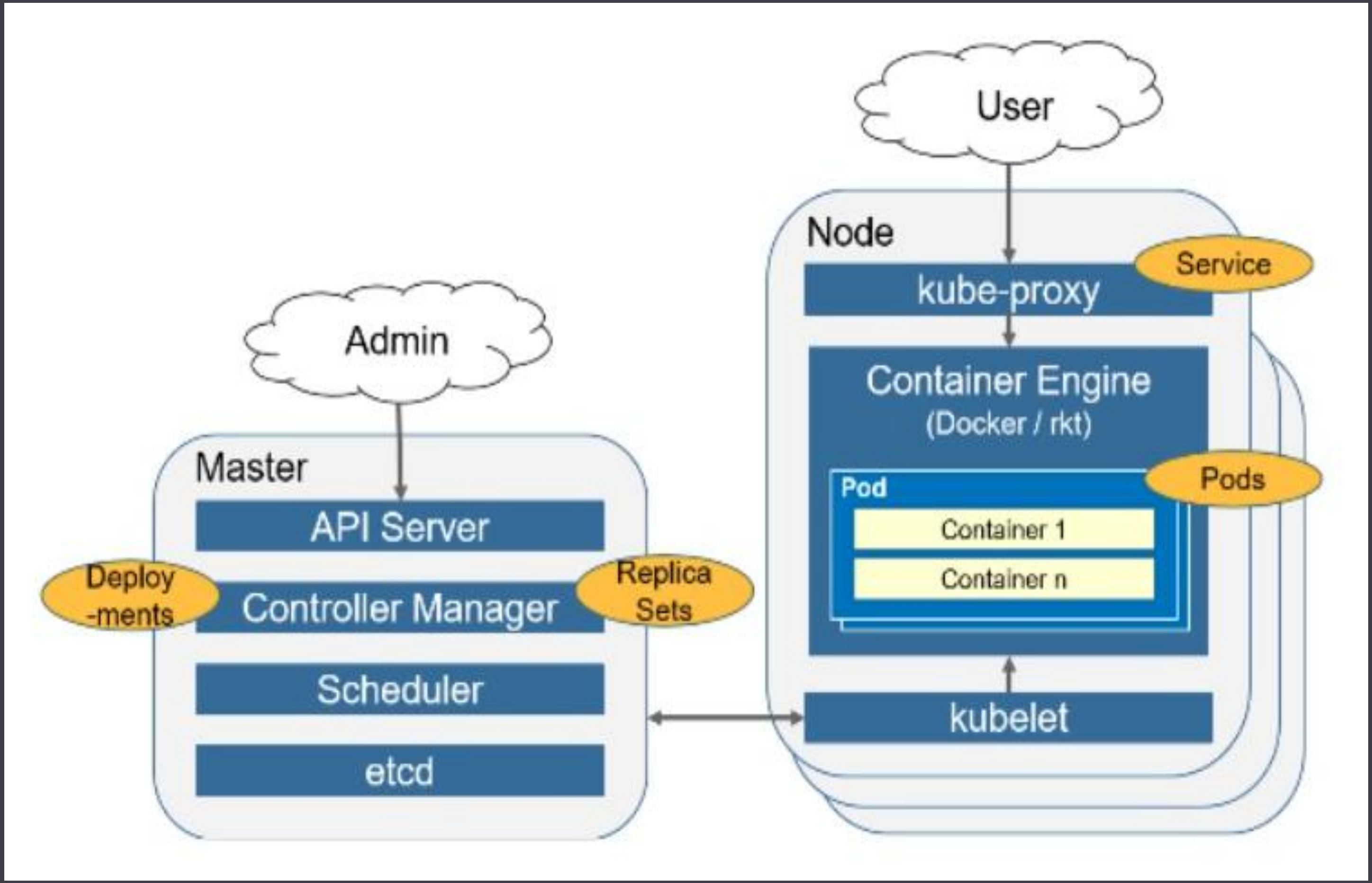
100% Open source, part of CNCF, written in Go

What is Kubernetes

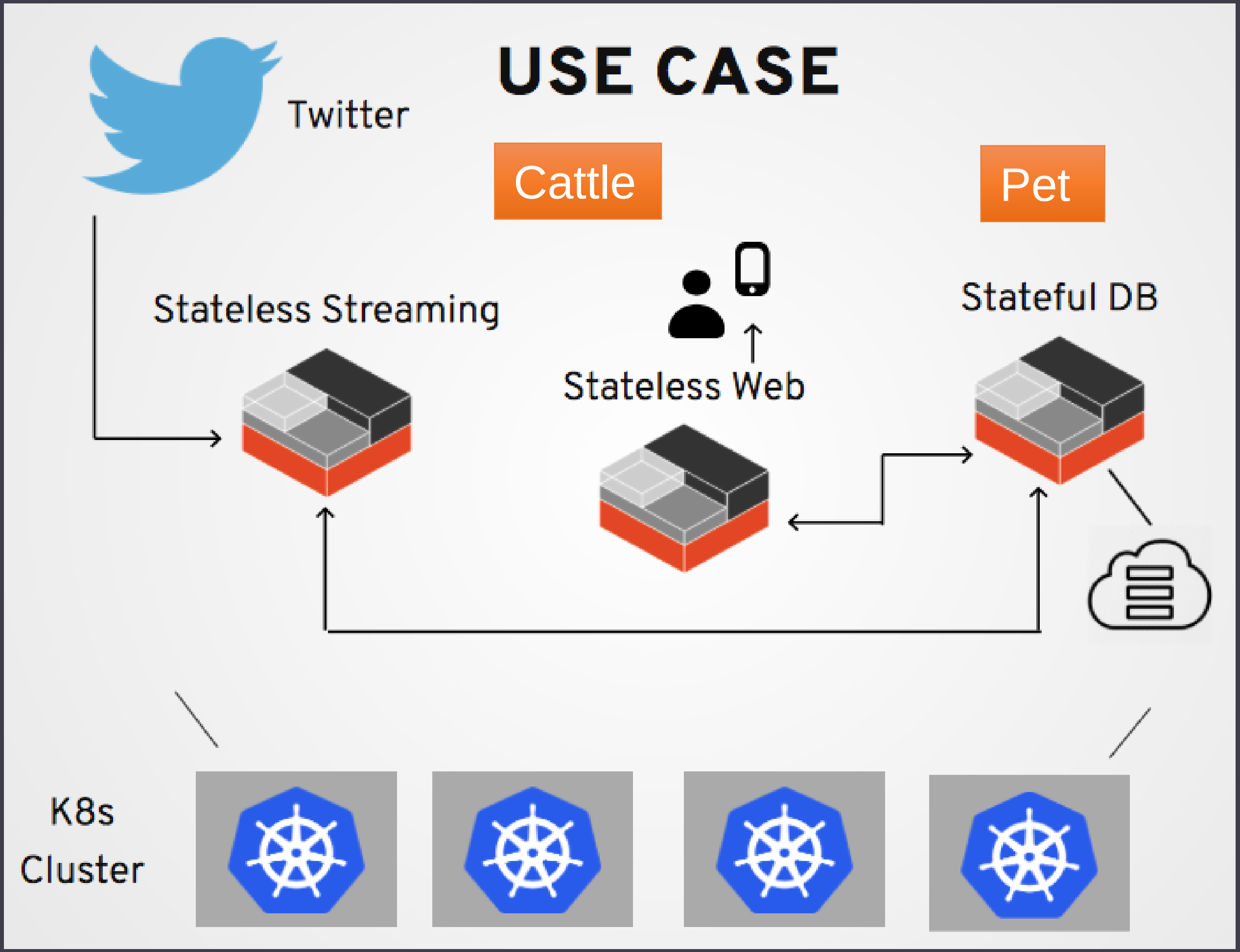
- **Services** are an abstraction for a logical set of Pods.
- **Pods** are the smallest deployable units of computing.
- **Deployments** provide declarative updates for Pods and RCs.
- **Replica Sets** ensure specified number of Pods are running.
- **Labels** are key/value pairs attached to objects used for identification.



What is Kubernetes



What is Kubernetes



Run Stateful Service on Kubernetes

- Stateless services are popular in containerized world because they are disposable and easy to be re-created with container images.

- Beyond stateless services like web server, users are increasingly deploying stateful services with containers to benefit from “build once, run anywhere” and to improve bare metal efficiency/utilization.

- These “pets” bring new requirements including longer life cycle, configuration dependencies and stateful failover. Container orchestration must address these needs to successfully deploy and scale apps.

Run Stateful Service on Kubernetes

Stateless

- Pod
- RC / Replica Set
- Service

→

Stateful

- Volume
- Persistent Volume

→

Clustered

- Pet Set
- Init Container
- Headless Service
- Dynamic Storage Provision

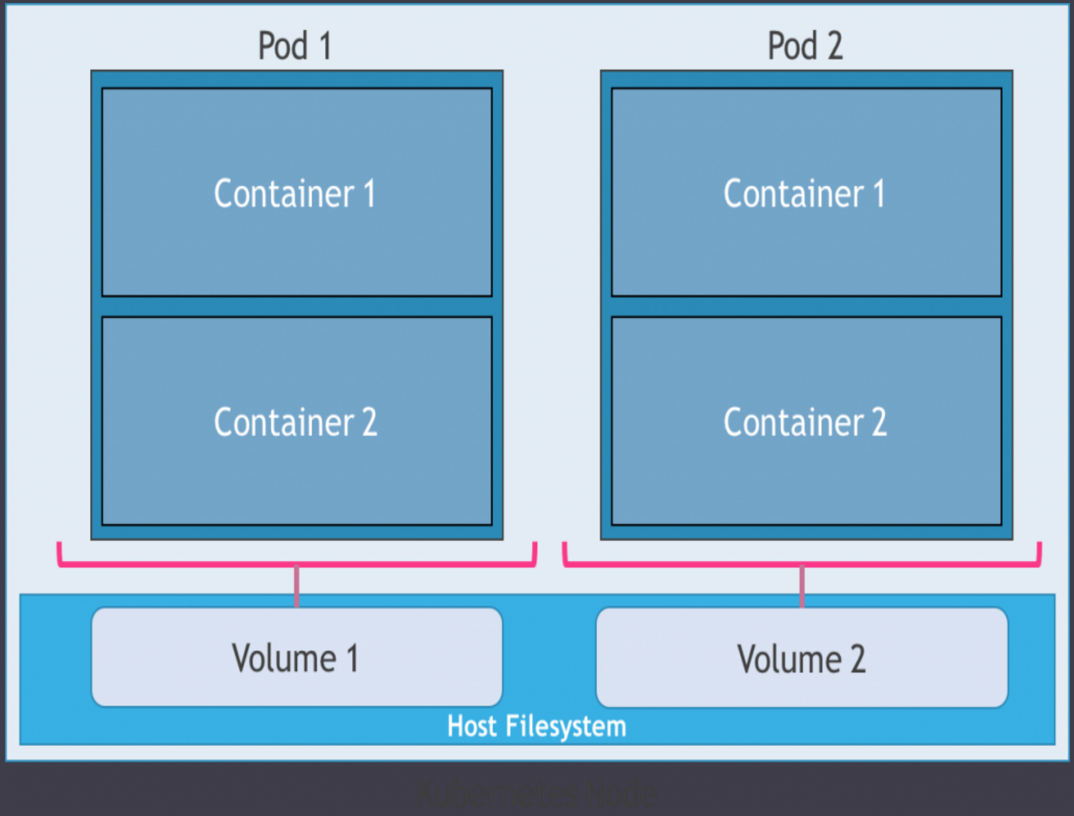
Agenda

- Background
  - What is Kubernetes?
  - Run stateful services on Kubernetes
- Kubernetes Stateful Service Features
  - Init Container
  - Pet Set
- Kubernetes Storage
  - Volume
  - Persistent Volume
  - Dynamic Storage Provision
- Run MySQL Cluster on Kubernetes
  - Galera MySQL Introduction
  - Deploy MySQL Cluster with PetSet
  - Cluster Operation on Kubernetes

Kubernetes Storage

- Node Based Volume
  - emptyDir
  - hostPath

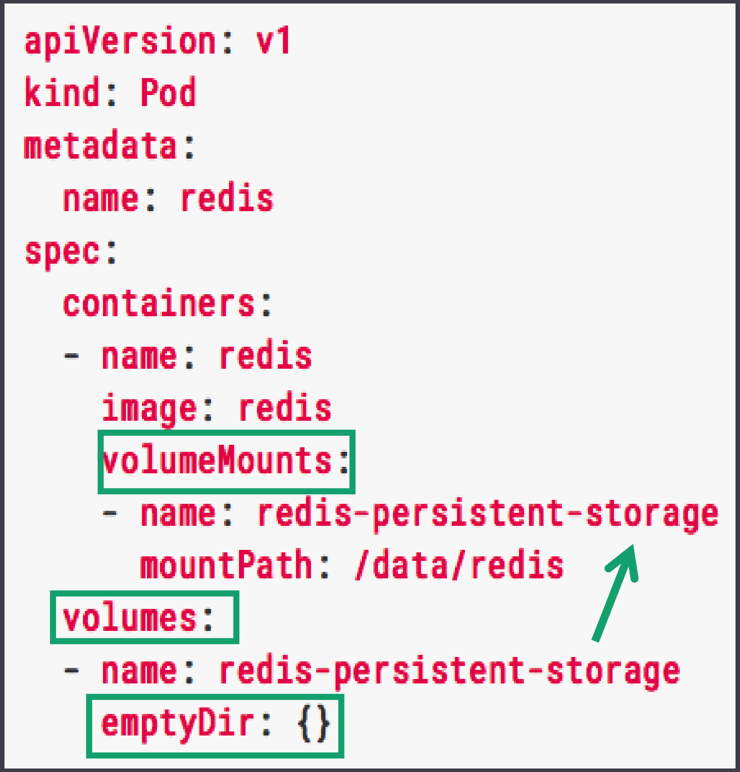
- Use Cases：
  - Store temp data
  - Data sharing among containers



时速云
tenxcloud.com

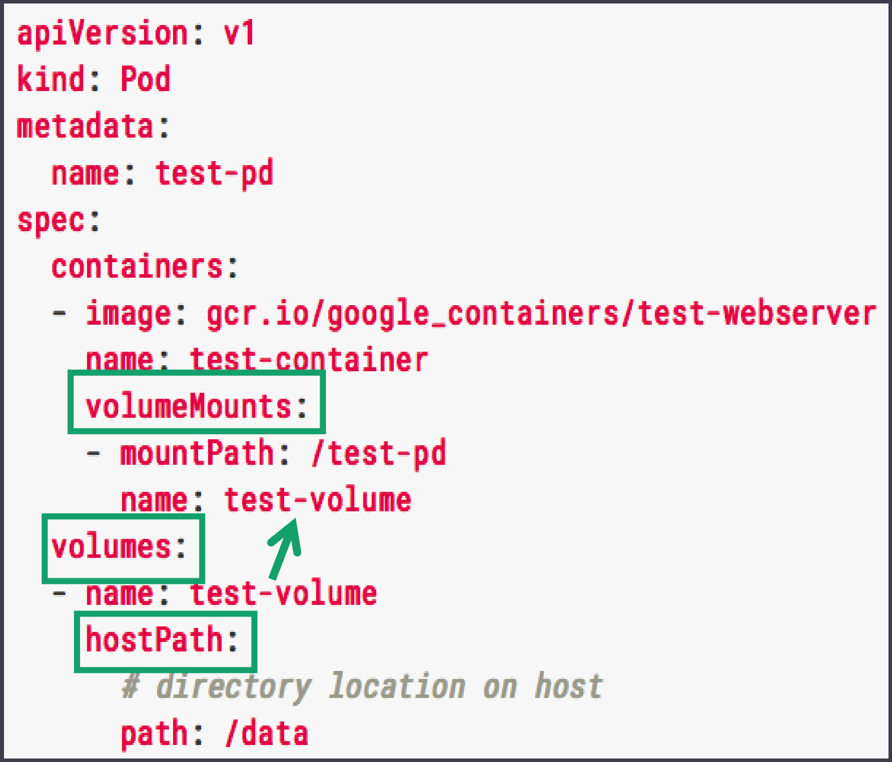
Kubernetes Storage

- emptyDir



```yaml
apiVersion: v1
kind: Pod
metadata:
  name: redis
spec:
  containers:
  - name: redis
    image: redis
    volumeMounts:
    - name: redis-persistent-storage
      mountPath: /data/redis
  volumes:
  - name: redis-persistent-storage
    emptyDir: {}
```

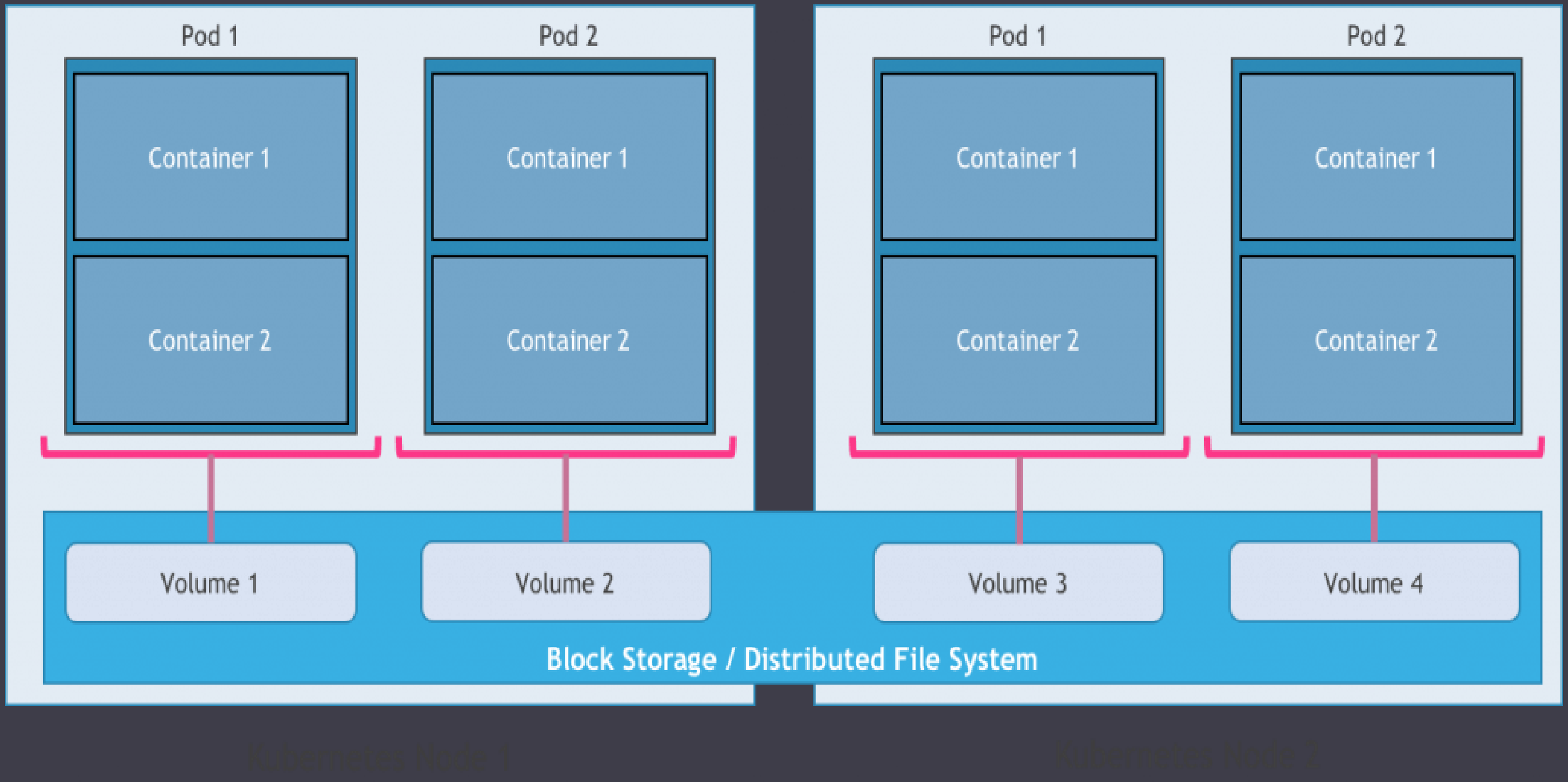
- hostPath



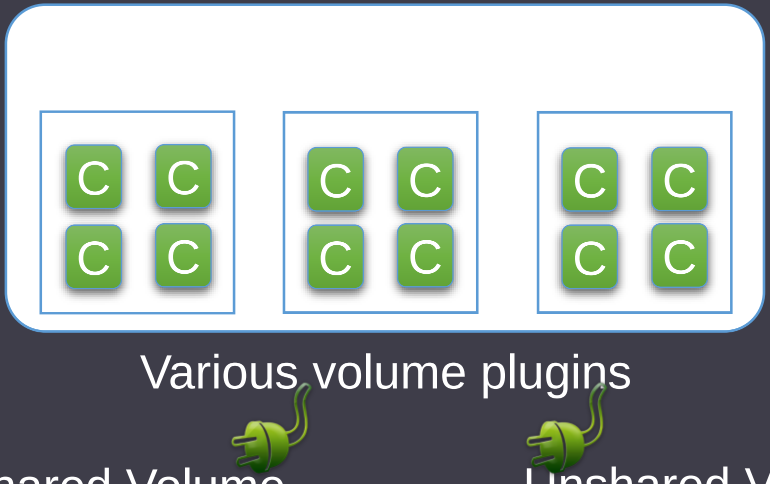
```yaml
apiVersion: v1
kind: Pod
metadata:
  name: test-pd
spec:
  containers:
  - image: gcr.io/google_containers/test-webserver
    name: test-container
    volumeMounts:
    - mountPath: /test-pd
      name: test-volume
  volumes:
  - name: test-volume
    hostPath:
      # directory location on host
      path: /data
```

Kubernetes Storage

- Cross Node Volume



Kubernetes Storage



Various volume plugins

Shared Volume

Unshared Volume

• nfs	• emptyDir
• hostPath	• hostPath
• glusterfs	• gcePersistentDisk
• flocker	• awsElasticBlockStore
• cephfs	• iscsi
• secret	• rbd
• vsphereVirtualDisk	• azureFileVolume

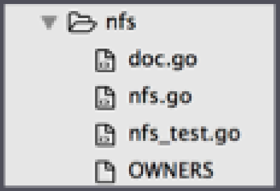
Kubernetes Storage

- Add your volume plugin

1. Register in kubelet entry
   kubelet/app/plugins.go

   allPlugins = append(allPlugins, customizedPlugin.ProbeVolumePlugins()...)

2. Implement it in the package below:
   pkg/volume/<your_plugin>, interface can refer to volume.go



```
▼ nfs
    doc.go
    nfs.go
    nfs_test.go
    OWNERS
```

- ProbeVolumePlugins
- Init
- CanSupport
- SetUp
- TearDown
- …

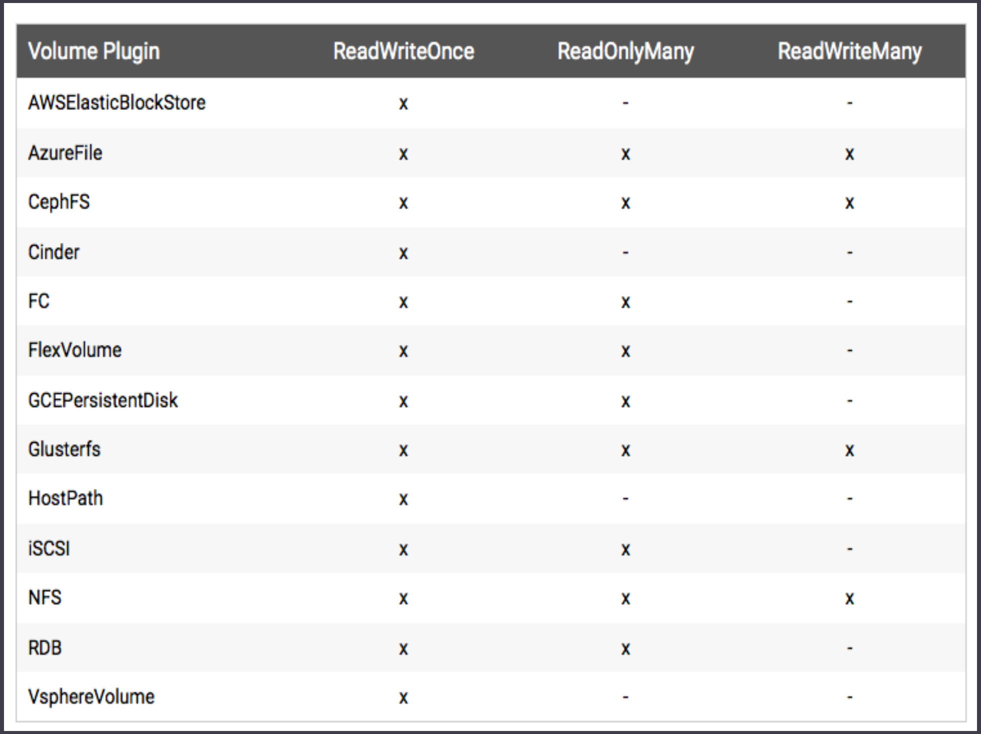
3. Update API Spec

Kubernetes Storage

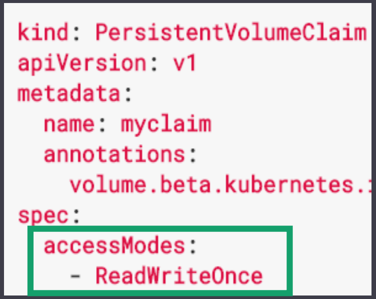
- Persistent Volume & Persistent Volume Claim
  - PV/PVC abstracts details of how storage is provided from how it is consumed.
  - PV/PVC are API resources.
  - PVs are volume plugins like Volumes, but have a lifecycle independent of any individual pod that uses the PV.

Kubernetes Storage

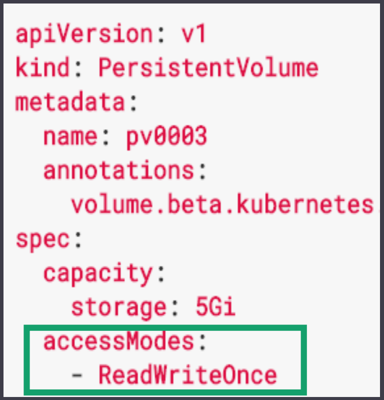
- PV Access Modes



Volume Plugin	ReadWriteOnce	ReadOnlyMany	ReadWriteMany
AWSElasticBlockStore	x	-	-
AzureFile	x	x	x
CephFS	x	x	x
Cinder	x	-	-
FC	x	x	-
FlexVolume	x	x	-
GCEPersistentDisk	x	x	-
Glusterfs	x	x	x
HostPath	x	-	-
iSCSI	x	x	-
NFS	x	x	x
RDB	x	x	-
VsphereVolume	x	-	-



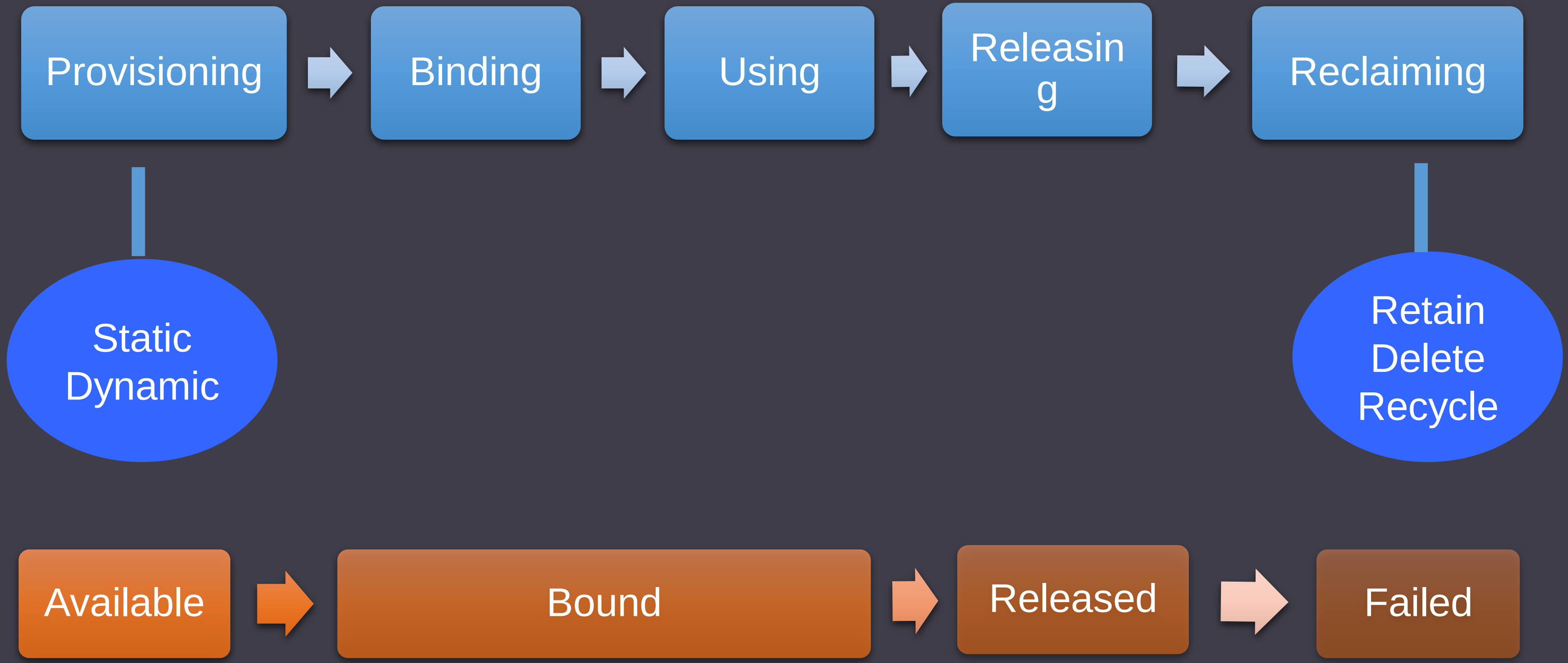
```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: myclaim
  annotations:
    volume.beta.kubernetes.
spec:
  accessModes:
    - ReadWriteOnce
```



```
apiVersion: v1
kind: PersistentVolume
metadata:
  name: pv0003
  annotations:
    volume.beta.kubernetes
spec:
  capacity:
    storage: 5Gi
  accessModes:
    - ReadWriteOnce
```

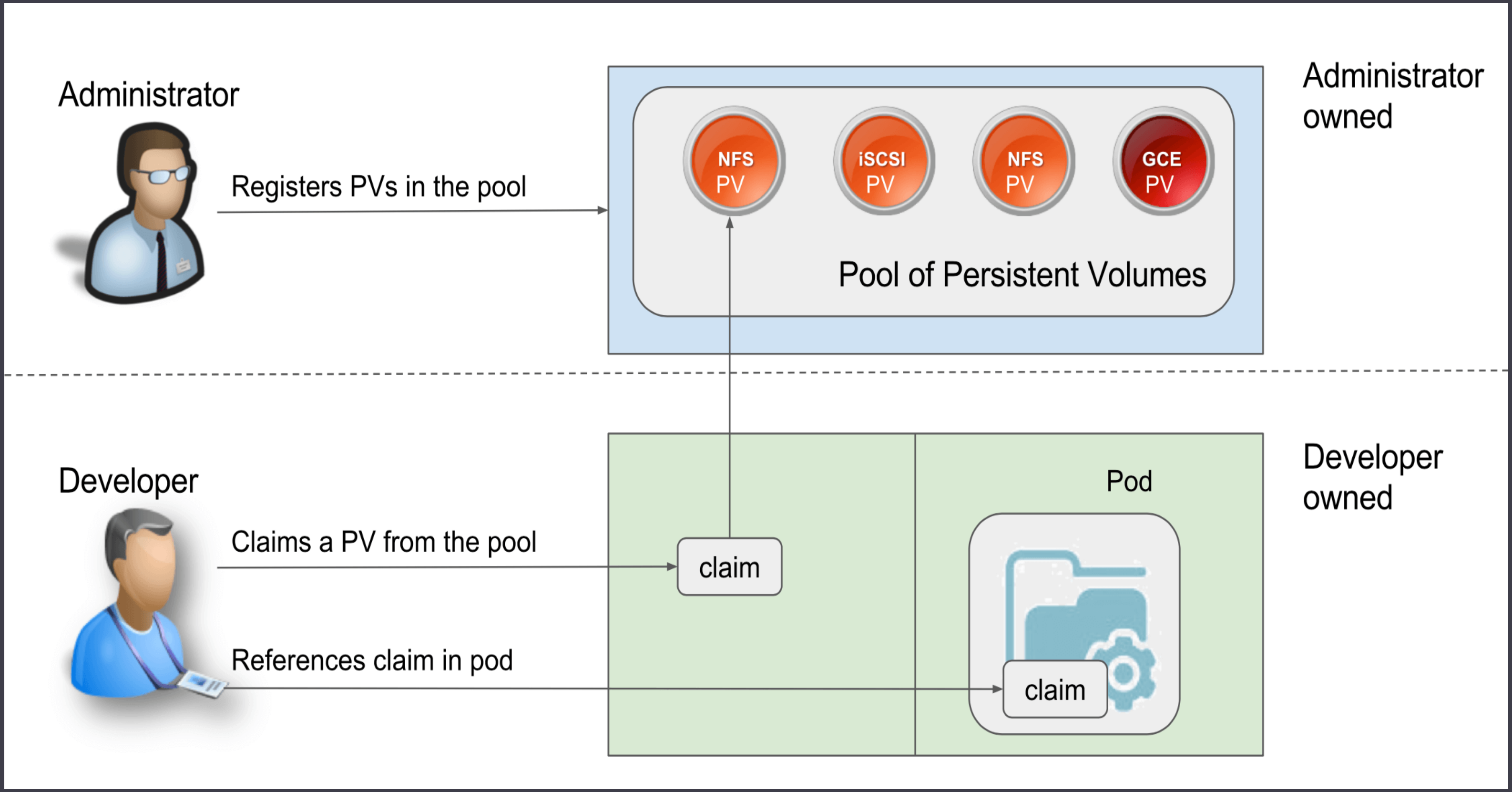
Kubernetes Storage

- PV Lifecycle



Kubernetes Storage

- PV Provision & Binding Process



Kubernetes Storage

- Dynamic Storage Provision

```
kind: StorageClass
apiVersion:
storage.k8s.io/v1beta1
metadata:
  name: slow
provisioner: kubernetes.io/gce-
pd
parameters:
  type: pd-standard
```

```
kind: StorageClass
apiVersion:
storage.k8s.io/v1beta1
metadata:
  name: fast
provisioner:
kubernetes.io/gce-pd
parameters:
  type: pd-ssd
```

Kubernetes Storage

- Dynamic Storage Provision

```
"kind": "PersistentVolumeClaim",
  "apiVersion": "v1",
  "metadata": {
    "name": "claim1",
    "annotations": {

"volume.beta.kubernetes.io/storage-
class": "fast"
    }
  },
```

```
"spec": {
    "accessModes": [
      "ReadWriteOnce"
    ],
    "resources": {
      "requests": {
        "storage": "30Gi"
      }
    }
  }
```

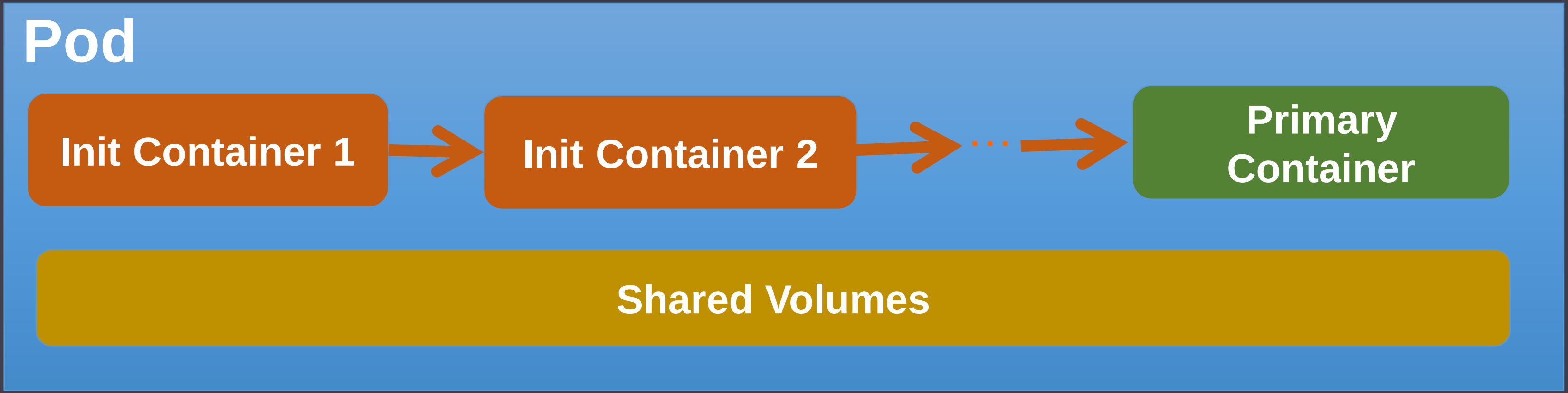
This claim will result in an SSD-like Persistent Disk being automatically provisioned. When the claim is deleted, the volume will be destroyed.

Agenda

- Background
  - What is Kubernetes?
  - Run stateful services on Kubernetes
- Kubernetes Storage
  - Volume
  - Persistent Volume
  - Dynamic Storage Provision
- Kubernetes Stateful Service Features
  - Init Container
  - Pet Set
- Run MySQL Cluster on Kubernetes
  - Galera MySQL Introduction
  - Deploy MySQL Cluster with PetSet
  - Cluster Operation on Kubernetes

Init Container

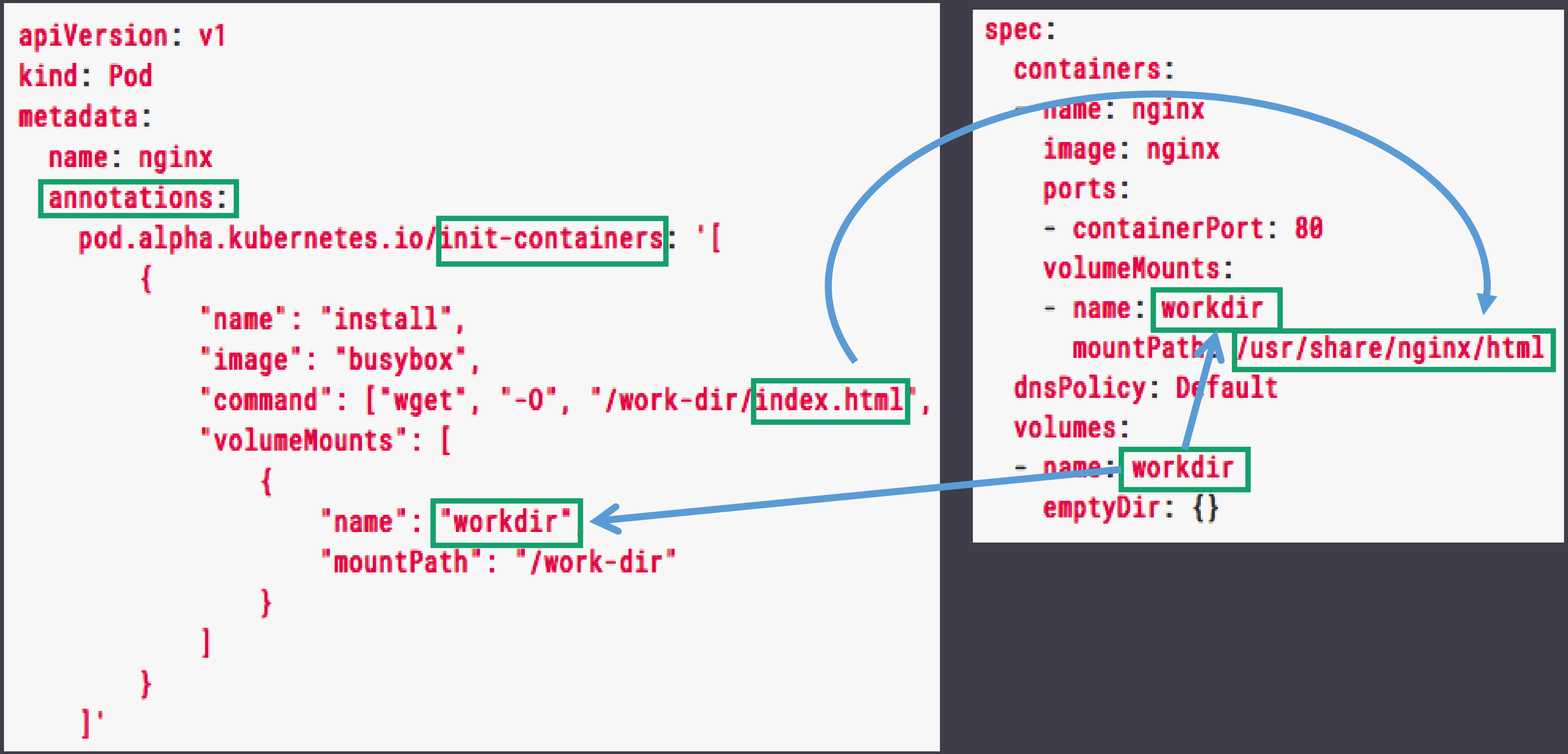
- What is Init Container
  - Sequentially executed containers in a pod
  - Initialize shared volumes
  - Always run to completion
  - Used on a pod, replica set, deployment, daemon set, pet set or job.



Init Container

- Why Init Container
    - Waiting for other components to be available
    - Performing configuration
    - Registering the pod into a central database
    - Downloading application dependencies
    - …

Init Container

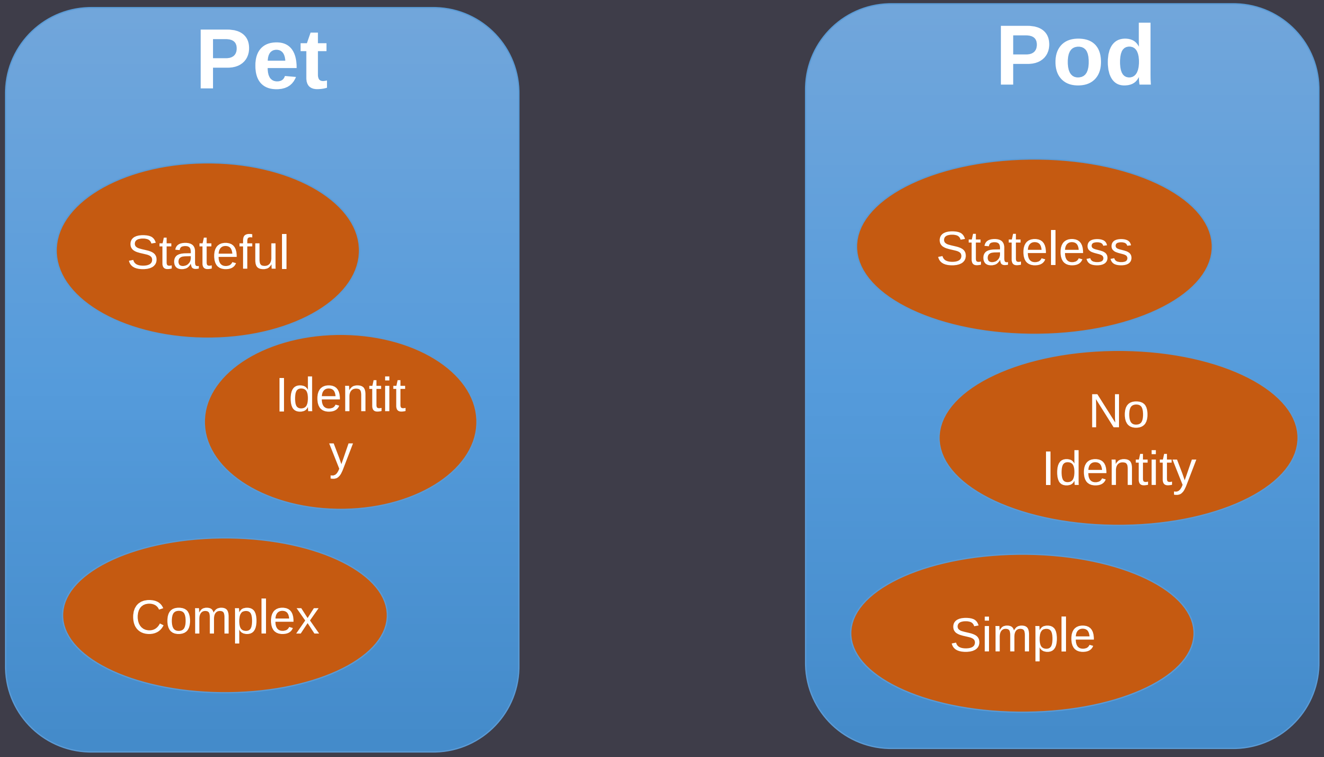


```
apiVersion: v1
kind: Pod
metadata:
  name: nginx
  annotations:
    pod.alpha.kubernetes.io/init-containers: '[
        {
            "name": "install",
            "image": "busybox",
            "command": ["wget", "-O", "/work-dir/index.html",
            "volumeMounts": [
                {
                    "name": "workdir",
                    "mountPath": "/work-dir"
                }
            ]
        }
    ]'
```

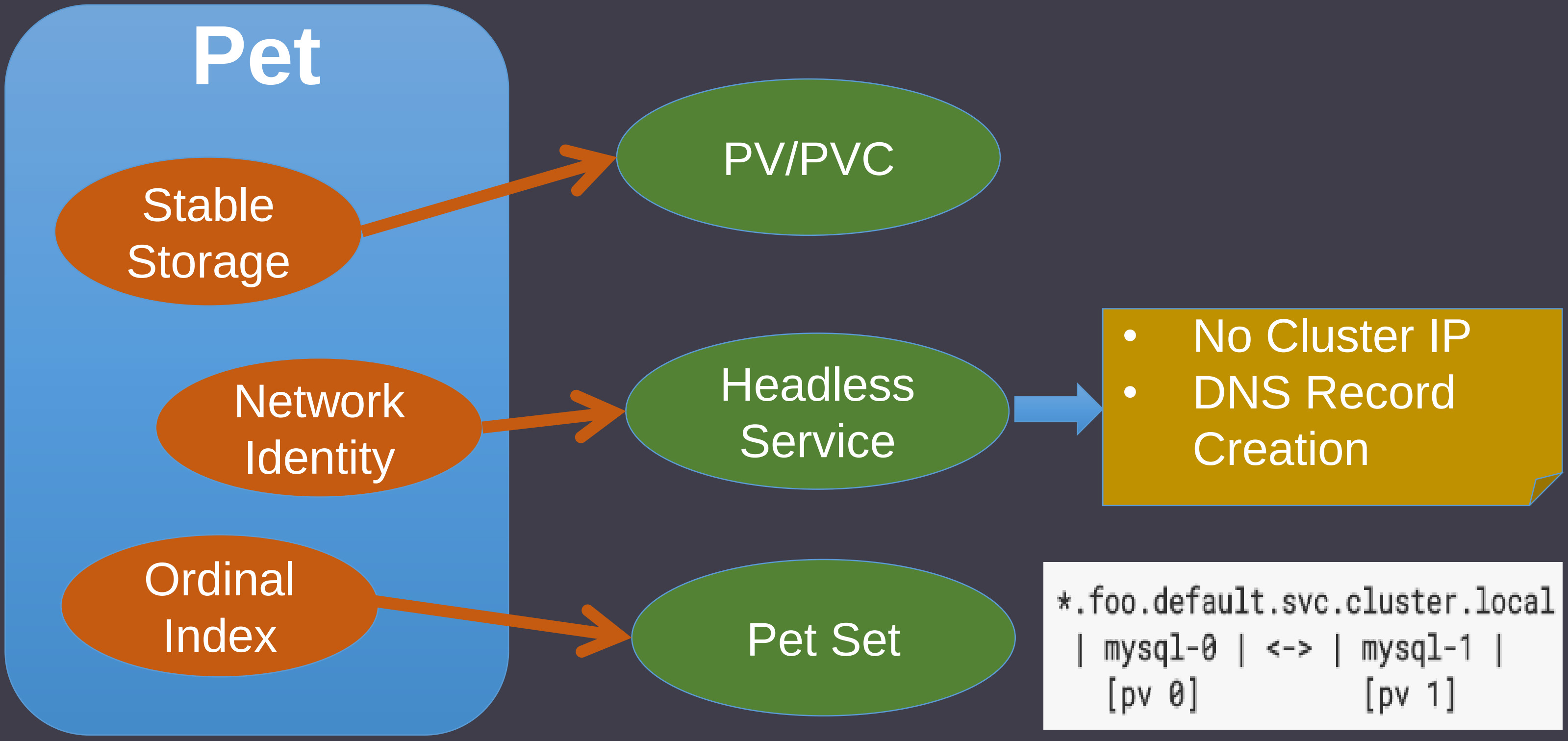
```
spec:
  containers:
  - name: nginx
    image: nginx
    ports:
    - containerPort: 80
    volumeMounts:
    - name: workdir
      mountPath: /usr/share/nginx/html
  dnsPolicy: Default
  volumes:
  - name: workdir
    emptyDir: {}
```

Pet Set

- Pet vs Pod



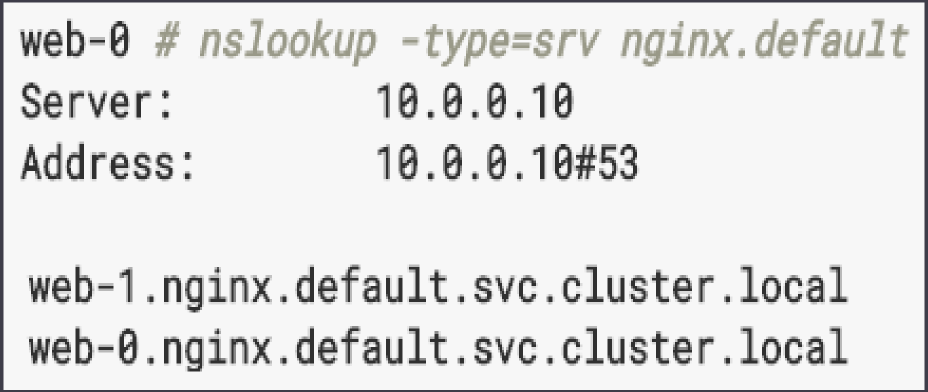
Pet Set



```
*.foo.default.svc.cluster.local
 | mysql-0 | <-> | mysql-1 |
   [pv 0]            [pv 1]
```

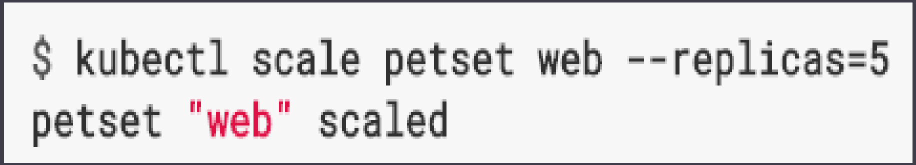
Pet Set

- PetSet Operations
  - Peer discovery
  - Scaling a PetSet
  - Image upgrades
  - Deleting a PetSet



```
web-0 # nslookup -type=srv nginx.default
Server:         10.0.0.10
Address:        10.0.0.10#53

 web-1.nginx.default.svc.cluster.local
 web-0.nginx.default.svc.cluster.local
```

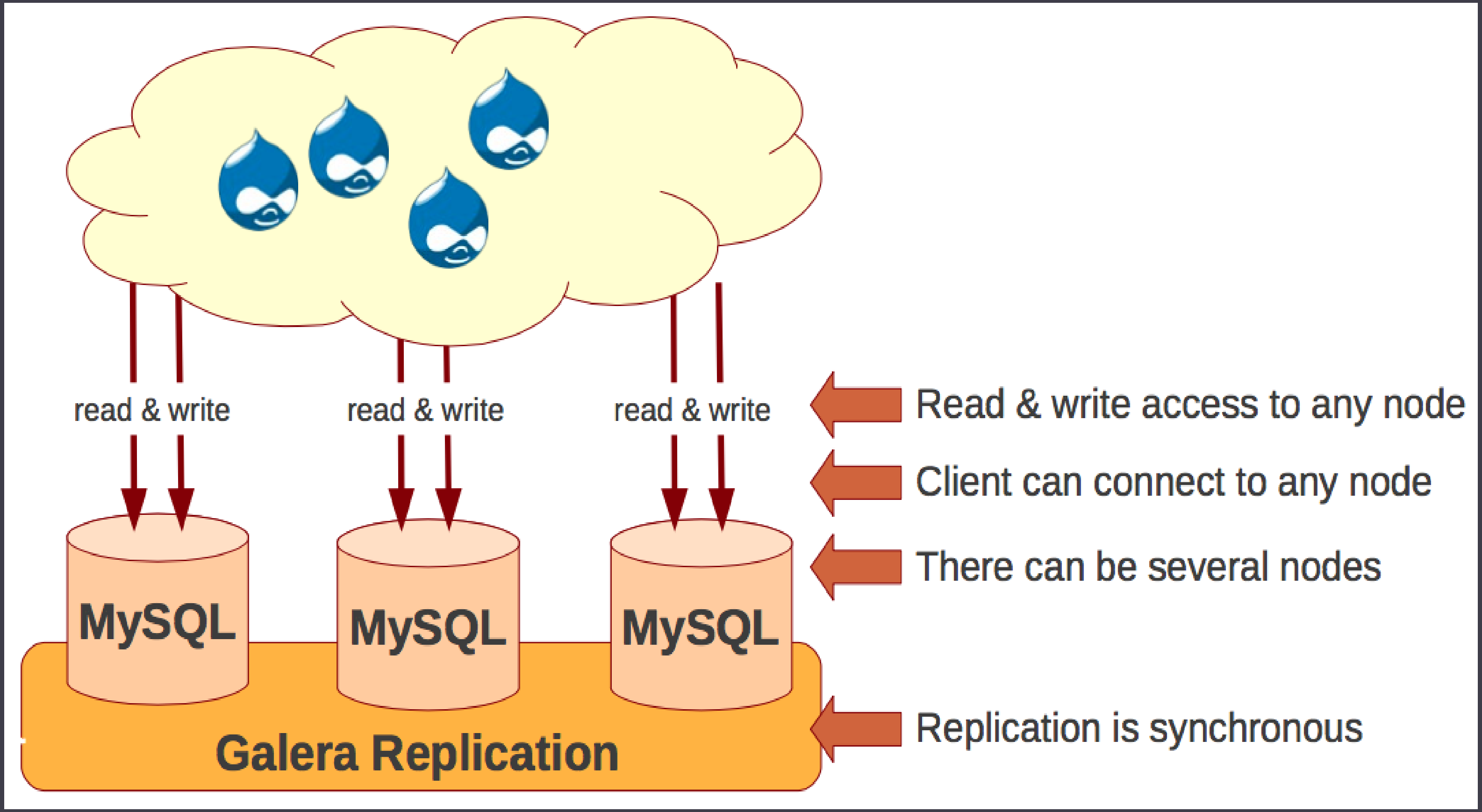


```
$ kubectl scale petset web --replicas=5
petset "web" scaled
```

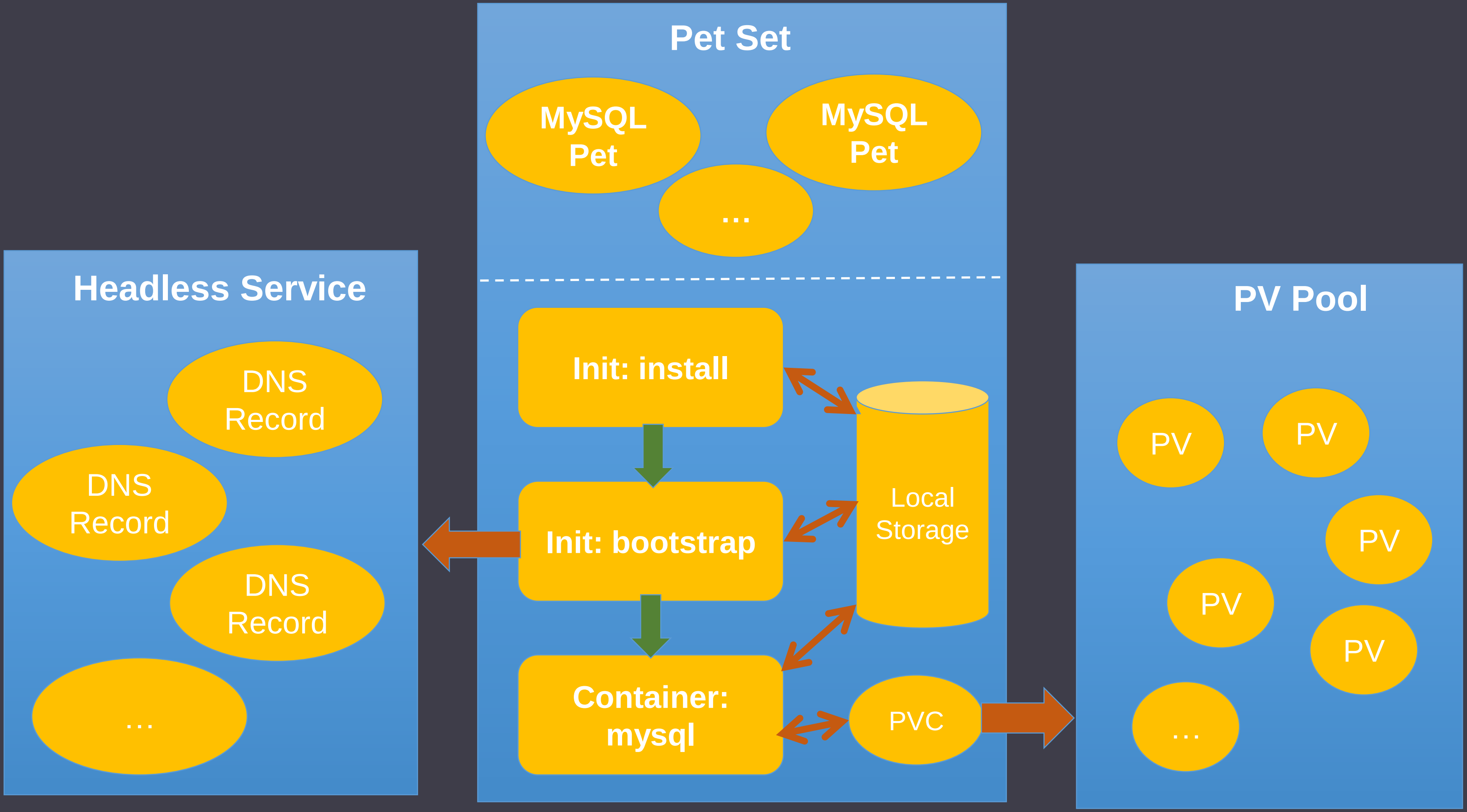
Agenda

- Background
  - What is Kubernetes?
  - Run stateful services on Kubernetes
- Kubernetes Stateful Service Features
  - Init Container
  - Pet Set
- Kubernetes Storage
  - Volume
  - Persistent Volume
  - Dynamic Storage Provision
- Run MySQL Cluster on Kubernetes
  - Galera MySQL Introduction
  - Deploy MySQL Cluster with PetSet
  - Cluster Operation on Kubernetes

Galera MySQL Introduction

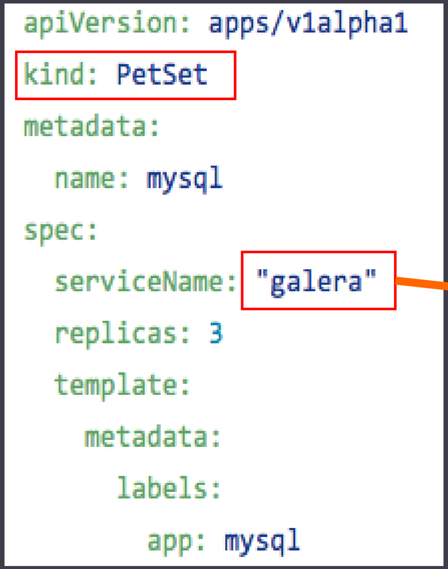

read & write
read & write
read & write
MySQL
MySQL
MySQL
Galera Replication
Read & write access to any node
Client can connect to any node
There can be several nodes
Replication is synchronous

Deploy MySQL Cluster with PetSet

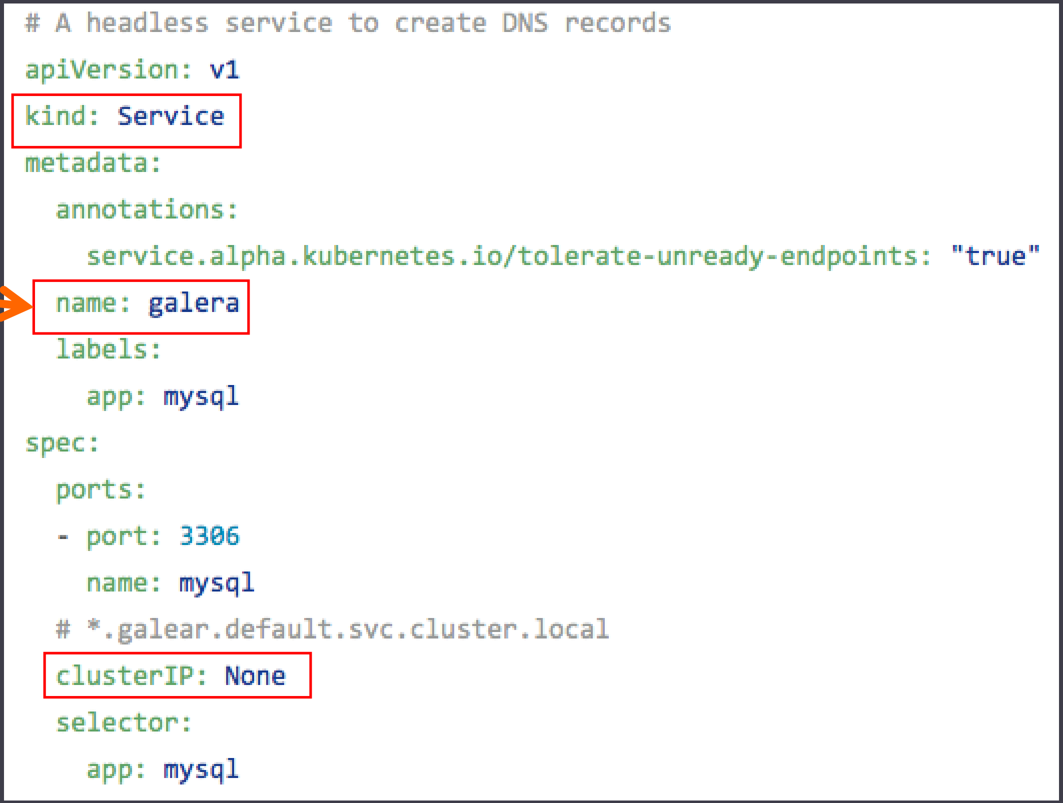


Deploy MySQL Cluster with PetSet

Headless Service & PetSet



```
apiVersion: apps/v1alpha1
kind: PetSet
metadata:
  name: mysql
spec:
  serviceName: "galera"
  replicas: 3
  template:
    metadata:
      labels:
        app: mysql
```

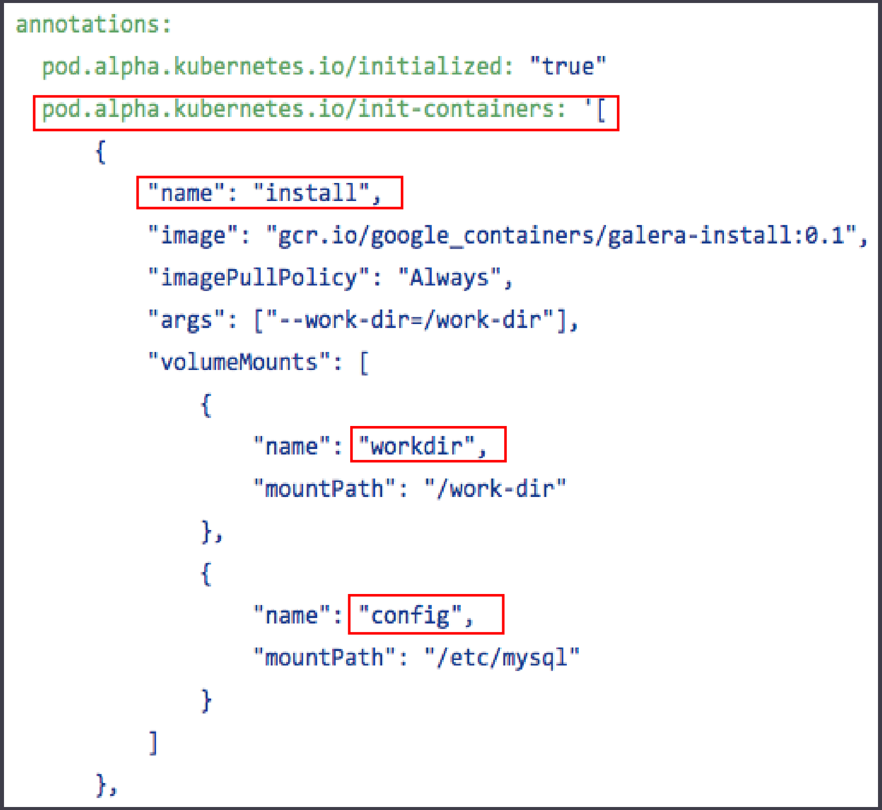


```
# A headless service to create DNS records
apiVersion: v1
kind: Service
metadata:
  annotations:
    service.alpha.kubernetes.io/tolerate-unready-endpoints: "true"
  name: galera
  labels:
    app: mysql
spec:
  ports:
  - port: 3306
    name: mysql
  # *.galear.default.svc.cluster.local
  clusterIP: None
  selector:
    app: mysql
```

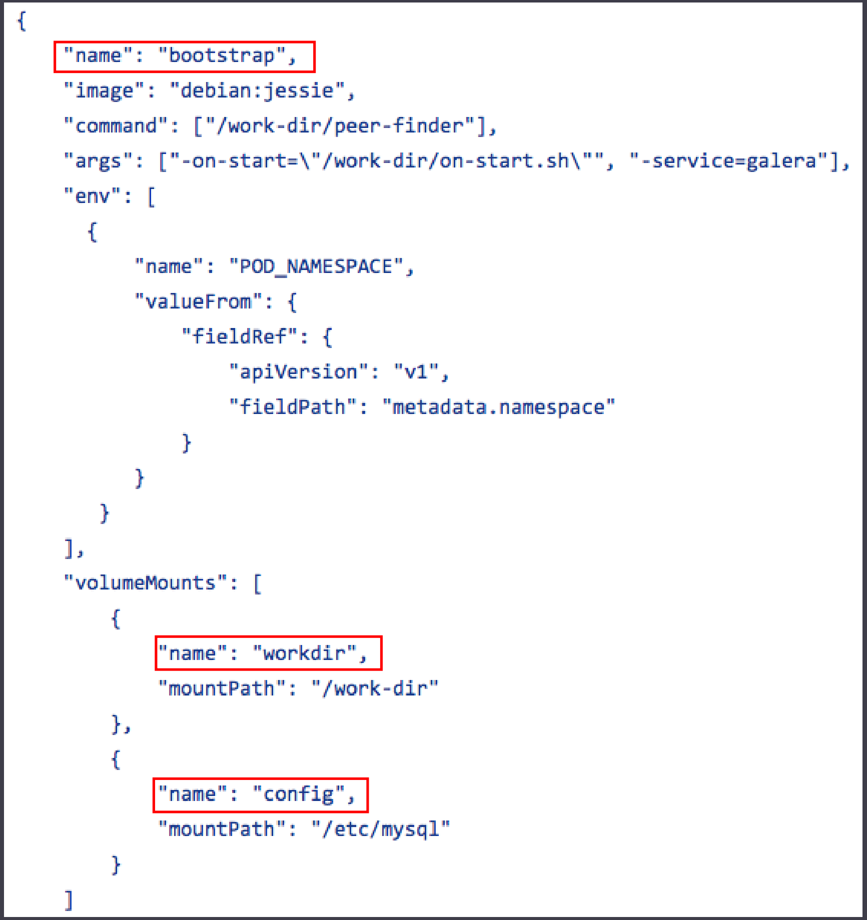
时速云
tenxcloud.com

Deploy MySQL Cluster with PetSet

Init Containers



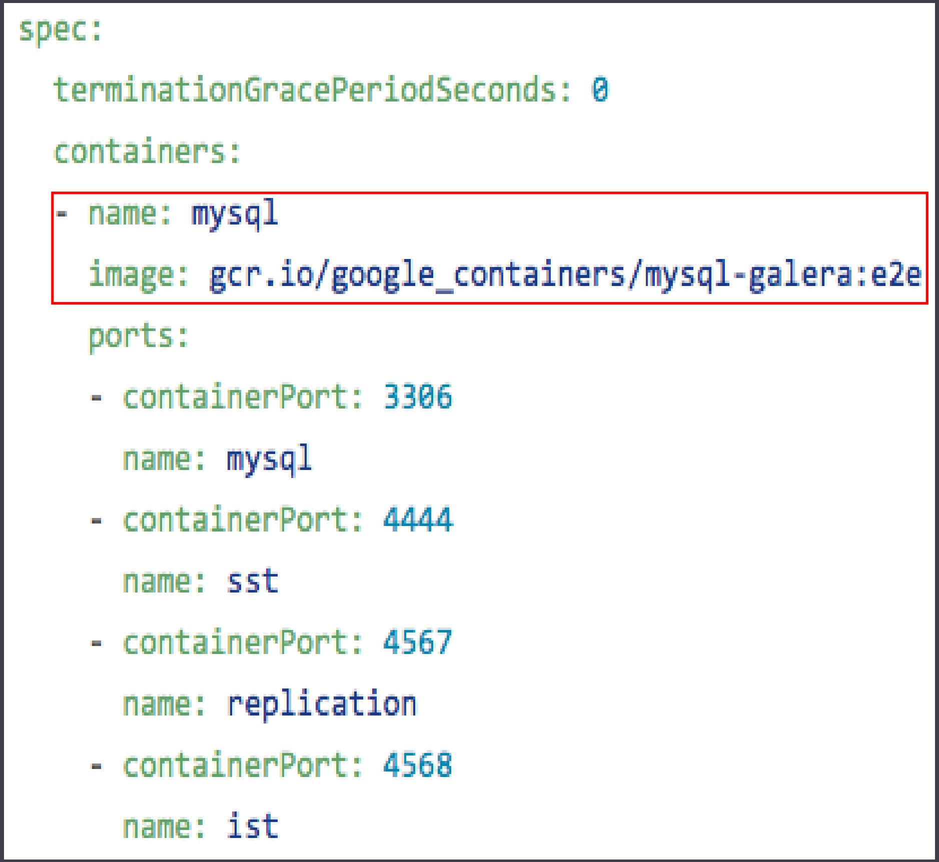
```
annotations:
  pod.alpha.kubernetes.io/initialized: "true"
  pod.alpha.kubernetes.io/init-containers: '[
      {
          "name": "install",
          "image": "gcr.io/google_containers/galera-install:0.1",
          "imagePullPolicy": "Always",
          "args": ["--work-dir=/work-dir"],
          "volumeMounts": [
              {
                  "name": "workdir",
                  "mountPath": "/work-dir"
              },
              {
                  "name": "config",
                  "mountPath": "/etc/mysql"
              }
          ]
      },
```



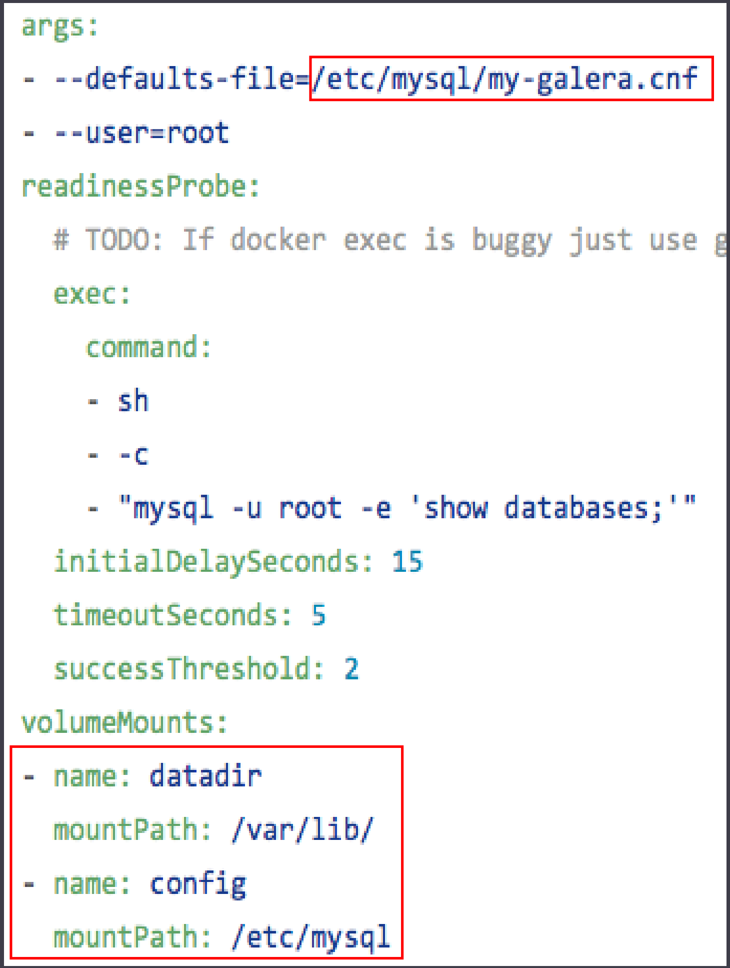
```
{
    "name": "bootstrap",
    "image": "debian:jessie",
    "command": ["/work-dir/peer-finder"],
    "args": ["-on-start=\"/work-dir/on-start.sh\"", "-service=galera"],
    "env": [
      {
          "name": "POD_NAMESPACE",
          "valueFrom": {
              "fieldRef": {
                  "apiVersion": "v1",
                  "fieldPath": "metadata.namespace"
              }
          }
      }
    ],
    "volumeMounts": [
        {
            "name": "workdir",
            "mountPath": "/work-dir"
        },
        {
            "name": "config",
            "mountPath": "/etc/mysql"
        }
    ]
```

Deploy MySQL Cluster with PetSet

Galera MySQL Container



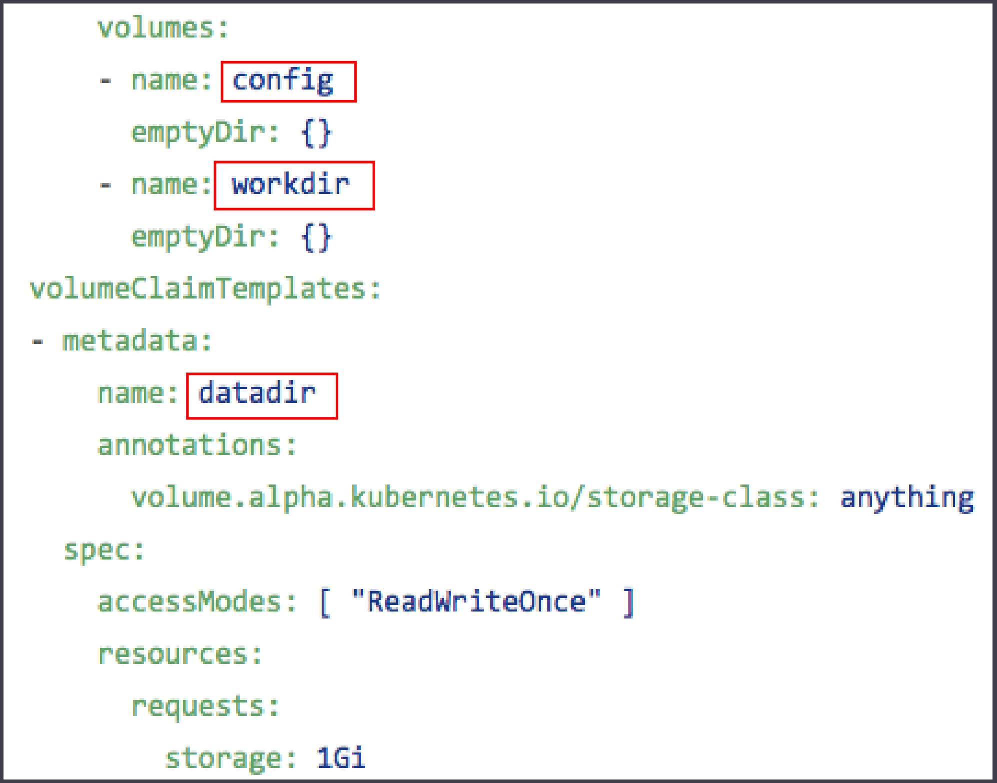
```
spec:
  terminationGracePeriodSeconds: 0
  containers:
  - name: mysql
    image: gcr.io/google_containers/mysql-galera:e2e
    ports:
    - containerPort: 3306
      name: mysql
    - containerPort: 4444
      name: sst
    - containerPort: 4567
      name: replication
    - containerPort: 4568
      name: ist
```



```
args:
- --defaults-file=/etc/mysql/my-galera.cnf
- --user=root
readinessProbe:
  # TODO: If docker exec is buggy just use g
  exec:
    command:
    - sh
    - -c
    - "mysql -u root -e 'show databases;'"
  initialDelaySeconds: 15
  timeoutSeconds: 5
  successThreshold: 2
volumeMounts:
- name: datadir
  mountPath: /var/lib/
- name: config
  mountPath: /etc/mysql
```

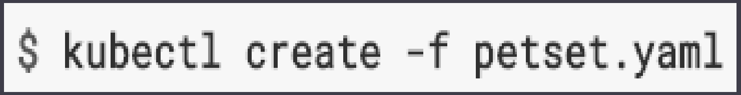
Deploy MySQL Cluster with PetSet

Volumes & PVC



```
    volumes:
    - name: config
      emptyDir: {}
    - name: workdir
      emptyDir: {}
volumeClaimTemplates:
- metadata:
    name: datadir
    annotations:
      volume.alpha.kubernetes.io/storage-class: anything
  spec:
    accessModes: [ "ReadWriteOnce" ]
    resources:
      requests:
        storage: 1Gi
```

Create Pet Set



```
$ kubectl create -f petset.yaml
```

Cluster Operation

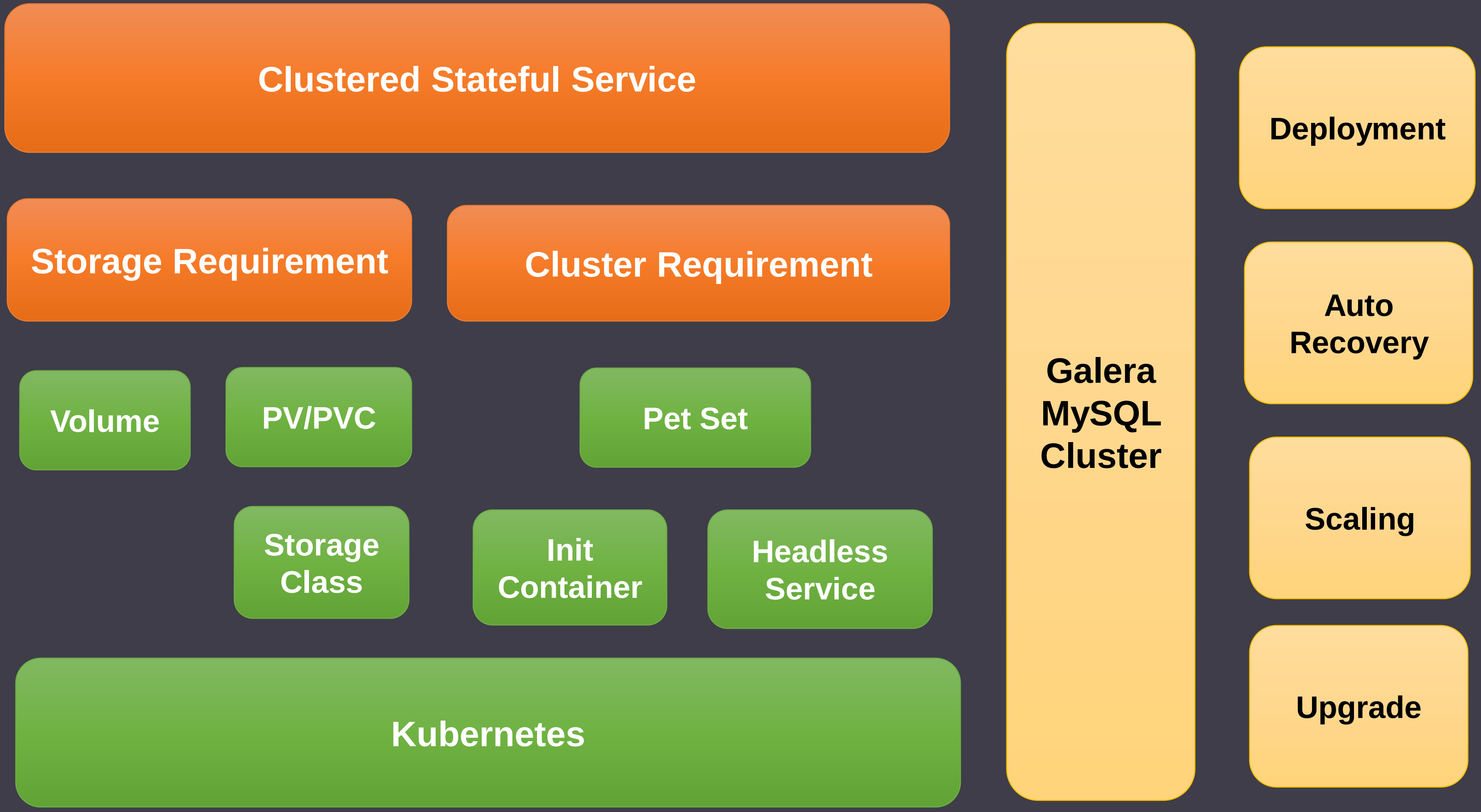
- Auto Recovery
  - The failed pets can be automatically recreated
  - The new created pet will use the data of the died one
- Scaling the Cluster
  - kubectl scale petset mysql --replicas=5
- Image Upgrades
  - Update the image field of any container in the podTemplate
  - Delete Pets one by one, the PetSet controller will recreate it with the new image

时速云
tenxcloud.com

Agenda

- Background
  - What is Kubernetes?
  - Run stateful services on Kubernetes
- Kubernetes Stateful Service Features
  - Init Container
  - Pet Set
- Kubernetes Storage
  - Volume
  - Persistent Volume
  - Dynamic Storage Provision
- Run MySQL Cluster on Kubernetes
  - Galera MySQL Introduction
  - Deploy MySQL Cluster with PetSet
  - Cluster Operation on Kubernetes

Summary


Clustered Stateful Service
Storage Requirement
Cluster Requirement
Volume
PV/PVC
Pet Set
Storage Class
Init Container
Headless Service
Kubernetes
Galera MySQL Cluster
Deployment
Auto Recovery
Scaling
Upgrade

Thanks！