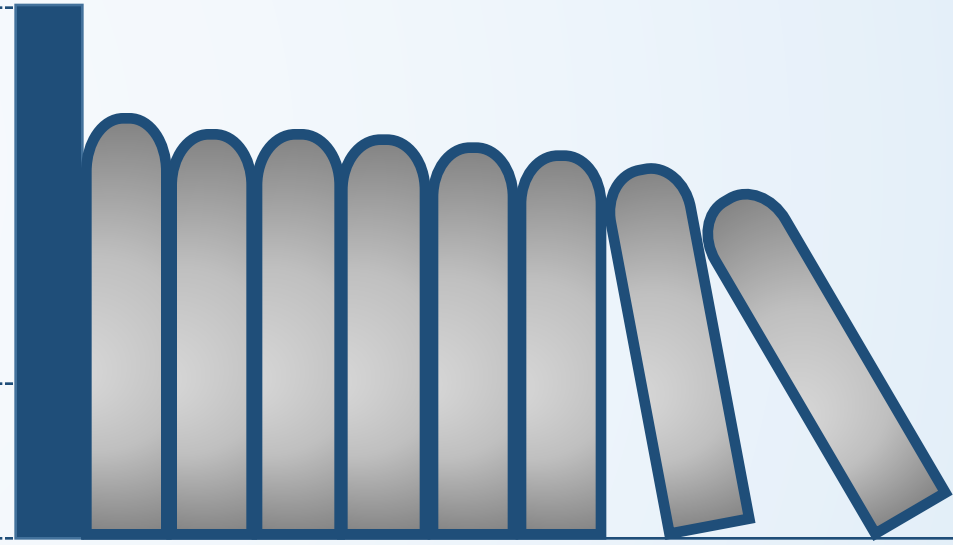
Kubernetes存储

主讲人：宋小金



目录

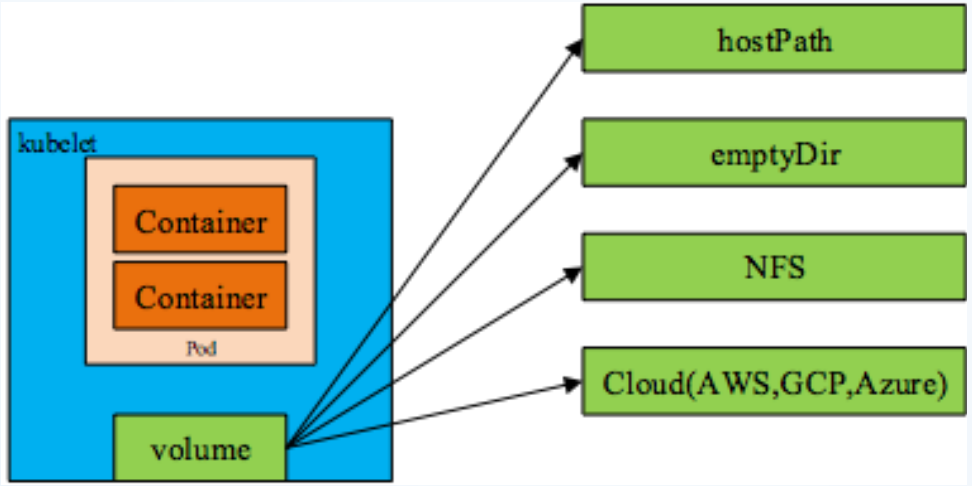
预期收获

- *了解Kubernetes的存储使用*

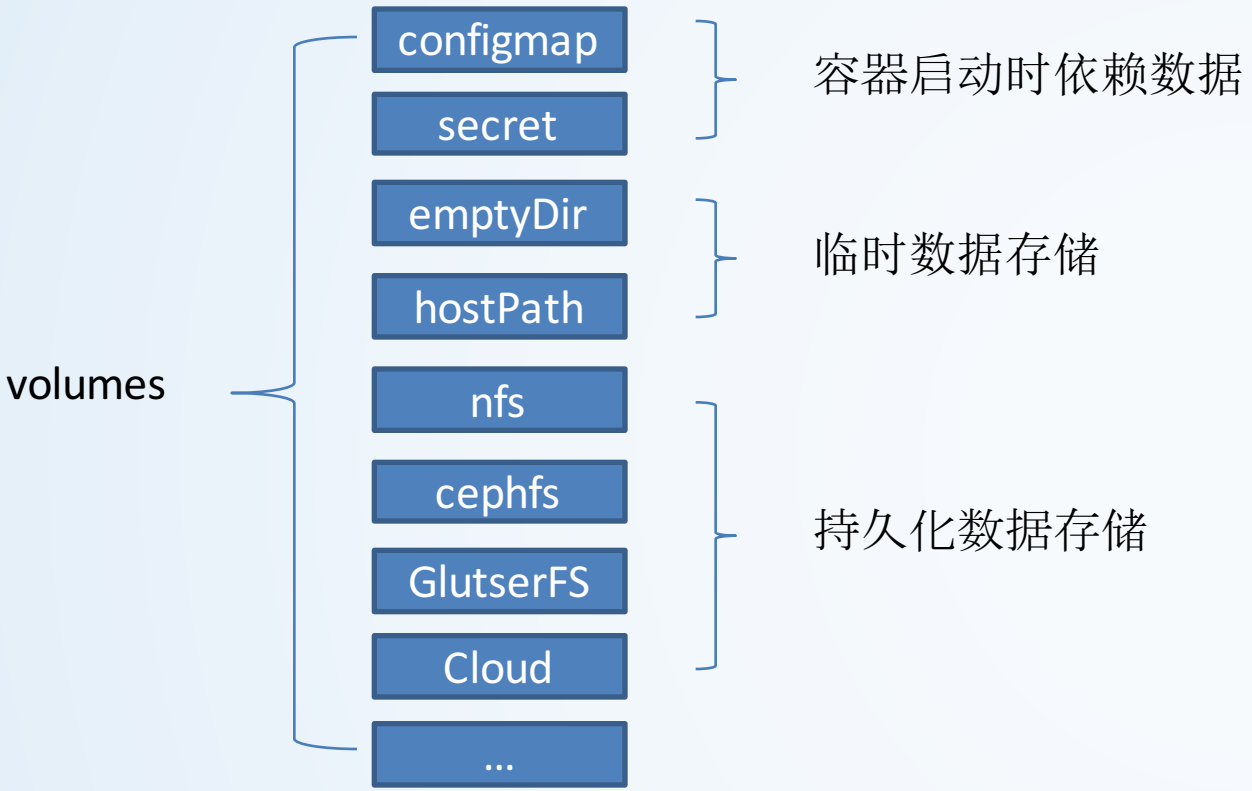
为何需要存储卷？

容器部署过程中一般有以下三种数据：
- 启动时需要的初始数据，可以是配置文件
- 启动过程中产生的临时数据，该临时数据需要多个容器间共享
- 启动过程中产生的持久化数据

以上三种数据都不希望在容器重启时就消失，存储卷由此而来，它可以根据不同场景提供不同类型的存储能力。



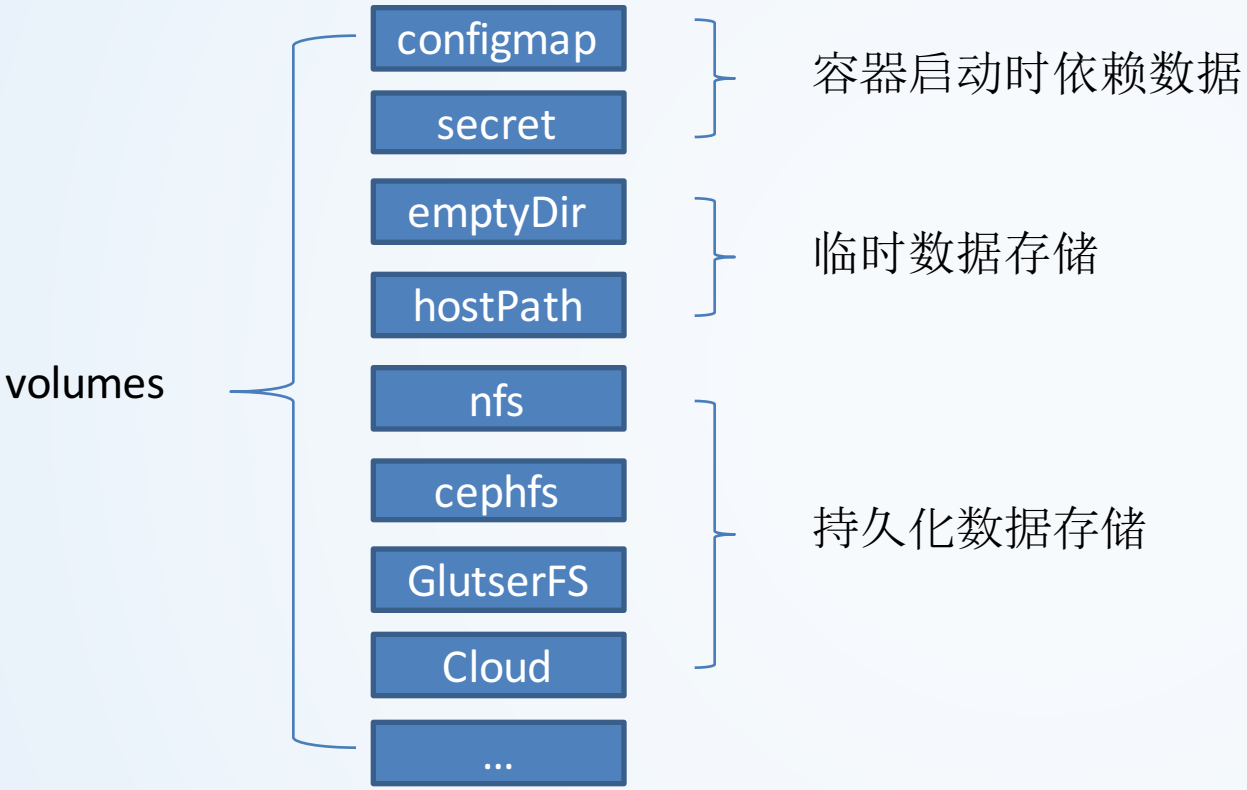
普通存储卷(volume)




```yaml
apiVersion: v1
kind: Pod
metadata:
  name: test-pd
spec:
  containers:
  - image: k8s.gcr.io/test-webserver
    name: test-container
    volumeMounts:
    - mountPath: /test-pd
      name: test-volume
  volumes:
  - name: test-volume
    hostPath:
      # directory location on host
      path: /data
```

存储没有单独资源对象，与Pod的生命周期一起

普通存储卷(volume)



存储没有单独资源对象，与Pod的生命周期一起

应用中使用普通卷

emptyDir

创建一个Pod时，会自动为emptyDir Volume在Pod所在的宿主机上分配一个空目录，当Pod被删除时，emptyDir Volume也会被删除

```
apiVersion: v1
kind: Pod
metadata:
  name: test-pd
spec:
  containers:
  - image: k8s.gcr.io/test-webserver
    name: test-container
    volumeMounts:
    - mountPath: /cache
      name: cache-volume
  volumes:
  - name: cache-volume
    emptyDir: {}
```

emptyDir使用场景

- Pod内多容器共享目录：比如一个Pod内的多个container共享一个目录（生产者-消费者）。
- 临时目录：某个应用运行需要临时目录，但该目录无需持久化保留。

通过*docker inspect*命令查看对应的docker container，emptyDir被随机分配到了宿主机的一个目录:

```
        "HostConfig": {
            "Binds": [
                "/var/lib/kubelet/pods/1bf009e9-495d-11e8-b24d-fa163ee6fdff/volumes/kubernetes.io~empty-dir/cache-volume:/cache"
```

应用中使用普通卷

hostPath

hostPath是在Pod上挂载宿主机的文件或目录，类似于*docker -v hostPath:containerPath*命令。

emptyDir被限制在一个Pod共享文件或目录，而hostPath可以实现跨Pod共享。

清理Volume

*docker volume prune*清理未被任何容器用到的Volume

hostPath优势

更好的持久化-本地

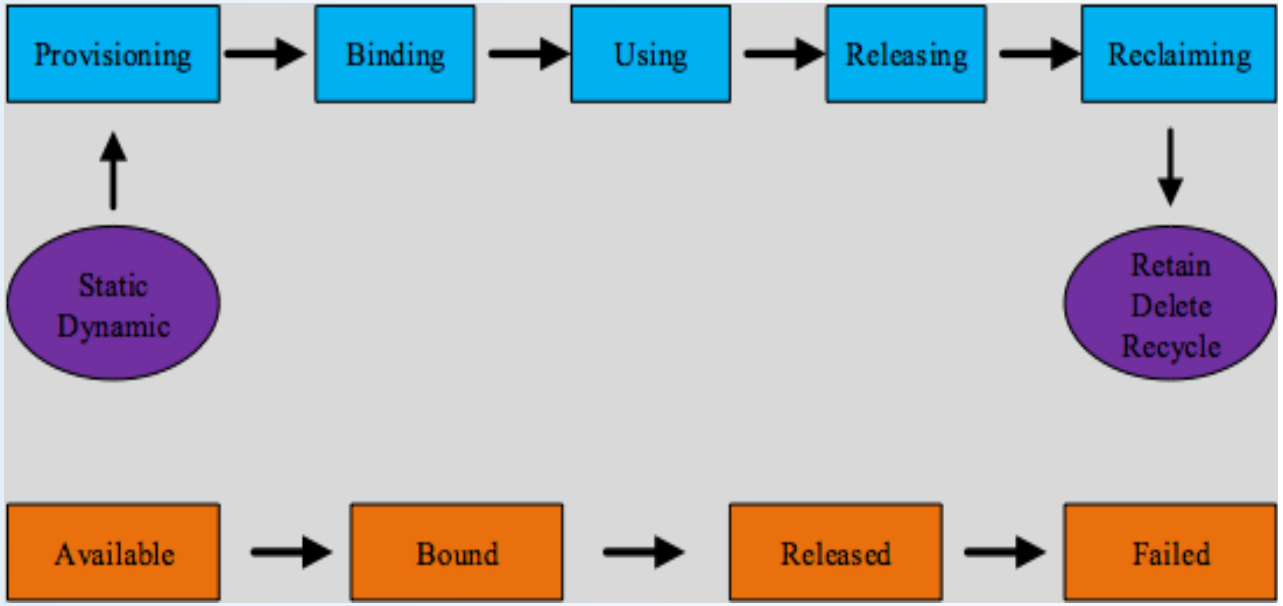
利用宿主机的高速文件系统

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: test-pd
  namespace: default
spec:
  containers:
  - name: busybox
    image: busybox
    command:
      - sleep
      - "3600"
    imagePullPolicy: IfNotPresent
    volumeMounts:
    - mountPath: /test-pd
      name: test-volume
  volumes:
  - name: test-volume
    hostPath:
      # directory location on host
      path: /data
      # this field is optional
      #type: Directory
  restartPolicy: Never
```

持久化存储卷(PersistentVolume)

存储系统与应用系统区分开，单独资源对象，它不直接和Pod发生关系，通过另一个资源对象PersistentVolumeClaim来绑定关联。

PV生命周期

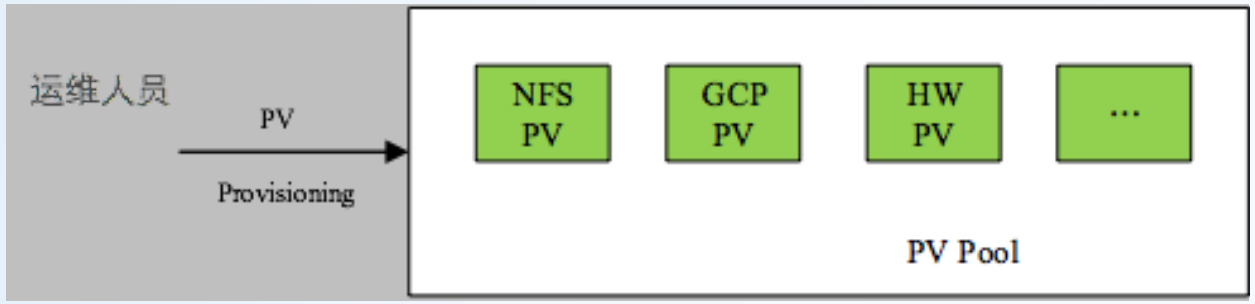


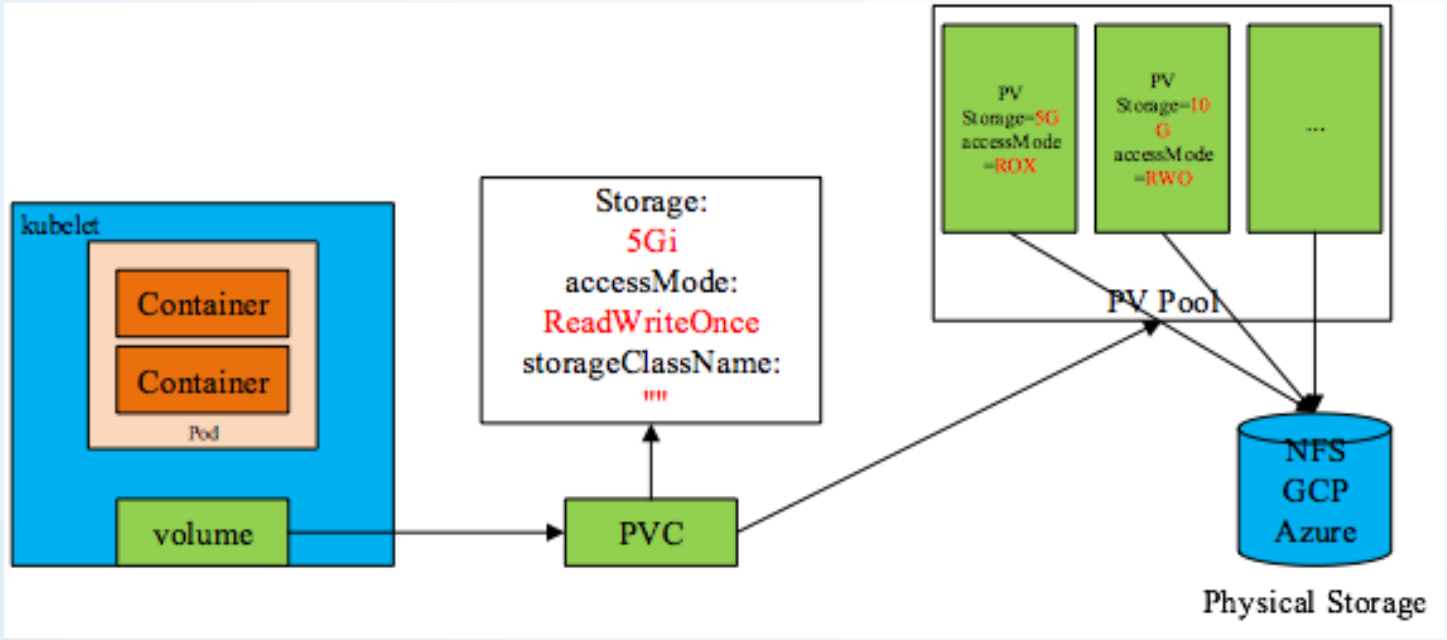

```yaml
apiVersion: v1
kind: PersistentVolume
metadata:
  name: pv0003
spec:
  capacity:
    storage: 5Gi
  volumeMode: Filesystem
  accessModes:
  - ReadWriteOnce
  persistentVolumeReclaimPolicy: Recycle
  storageClassName: slow
  mountOptions:
  - hard
  - nfsvers=4.1
  nfs:
    path: /tmp
    server: 172.17.0.2
```

持久化存储卷(PersistentVolume)

Provisioning：PV的预制创建有两种模式：静态模式和动态模式

静态模式：除创建PVC外，还需手动创建PV



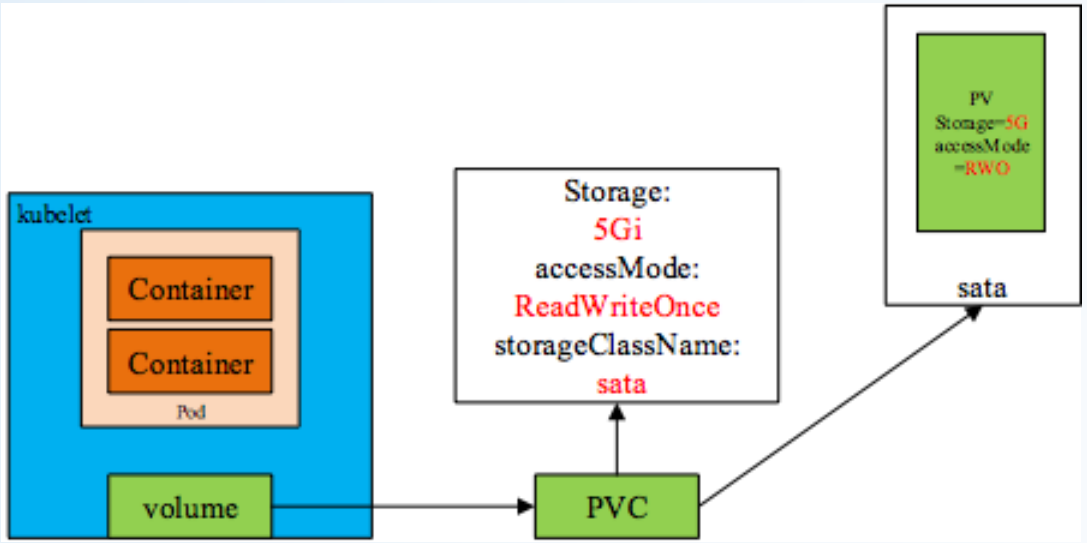



```yaml
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: mongodb-pv-claim
  labels:
    app: mongodb
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 5Gi

---
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: rsvp-db
spec:
  replicas: 1
  template:
    metadata:
      labels:
        appdb: rsvpdb
    spec:
      containers:
      - name: rsvpd-db
        image: mongo:3.3
        ports:
        - containerPort: 27017
        volumeMounts:
        - name : mongodb-persistent-storage
          mountPath : /data/db
      volumes:
      - name: mongodb-persistent-storage
        persistentVolumeClaim:
          claimName: mongodb-pv-claim
```

持久化存储卷(PersistentVolume)

动态模式：只需创建PVC，系统根据PVC自动创建PV



支持的自动创建存储类型

```yaml
kind: StorageClass
apiVersion: storage.k8s.io/v1
metadata:
  name: fast
provisioner: kubernetes.io/gce-pd
parameters:
  type: pd-ssd
  zone: us-east1-d
```

```yaml
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
 name: pvc-engineering
spec:
 accessModes:
  - ReadWriteMany
 resources:
   requests:
     storage: 10Gi
 storageClassName: fast
```

```yaml
kind: Pod
apiVersion: v1
metadata:
  name: mypod
spec:
  containers:
    - name: myfrontend
      image: nginx
      volumeMounts:
      - mountPath: "/var/www/html"
        name: mypd
  volumes:
    - name: mypd
      persistentVolumeClaim:
        claimName: pvc-engineering
```

https://docs.openshift.com/container-platform/3.9/install_config/storage_examples/storage_classes_dynamic_provisioning.html
https://kubernetes.io/docs/concepts/storage/storage-classes/#aws

课程回顾

已学知识要点

了解Kubernetes存储使用