Kubernetes存储概览

邢　舟　IBM开放技术研究院
xingzhou@cn.ibm.com
2017.6

IBM

Open by design™

议程

- K8s存储的主要应用场景
- K8s存储的设计原则
- K8s存储的主要模块
- K8s存储的社区开发
- 一个简单的例子

K8s存储的主要应用场景

- 应用程序 / 服务存储状态、数据存取等
- 应用程序 / 服务配置文件读取、秘钥配置等
- 不同应用程序间或者应用程序内进程间共享数据

K8s存储的主要应用场景——一个例子（1/4）

- 需求
  - 部署一个Nginx服务，并在/var/nginx-data目录下存储用户上传的数据
- 解决：
  - 使用AWS的弹性存储卷并将其挂载至K8s容器指定目录

K8s存储的主要应用场景——一个例子（2/4）

- Step 1
  - 创建一个AWS存储卷
  - 获取创建好的存储卷ID

- aws ec2 create-volume –availability-zone xxx –size xx …


```
{
        "AvailabilityZone": ”xxx”,
        "Encrypted": false,
        "VolumeType": "gp2”,
        "VolumeId": "vol-xxxxxxxxxxxxxxxxx”,
        "State": "creating”,
        "Iops": xxx,
        "SnapshotId": ””,
        "CreateTime": ”xxxxxxxxxxxx”,
        "Size": xx
}
```

K8s的主要应用场景——一个例子（3/4）

- Step 2
  - 创建一个包含nginx pod定义的yaml文件
  - 文件中包含存储卷在pod容器中的挂载位置
  - 以及K8s存储卷定义，其中使用到了Step1中所创建的存储卷的ID

```
apiVersion: v1
kind: Pod
metadata:
    name: nginx
spec:
    containers:
    - image: nginx
      name: nginx-server
      volumeMounts:
      - mountPath: /var/nginx-data
        name: data-volume
    volumes:
    - name: data-volume
      awsElasticBlockStore:
          volumeID: vol-xxxxxxxxxxxxxxxxx
          fsType: ext4
```

K8s存储的主要应用场景——一个例子（4/4）

- Step 3
  - 基于Step2中创建的yaml文件在K8s集群中创建nginx pod
  - 登录运行中的pod, 验证/var/nginx-data已经正确挂载

Credit:

http://leebriggs.co.uk/blog/2017/03/12/kubernetes-flexvolumes.html

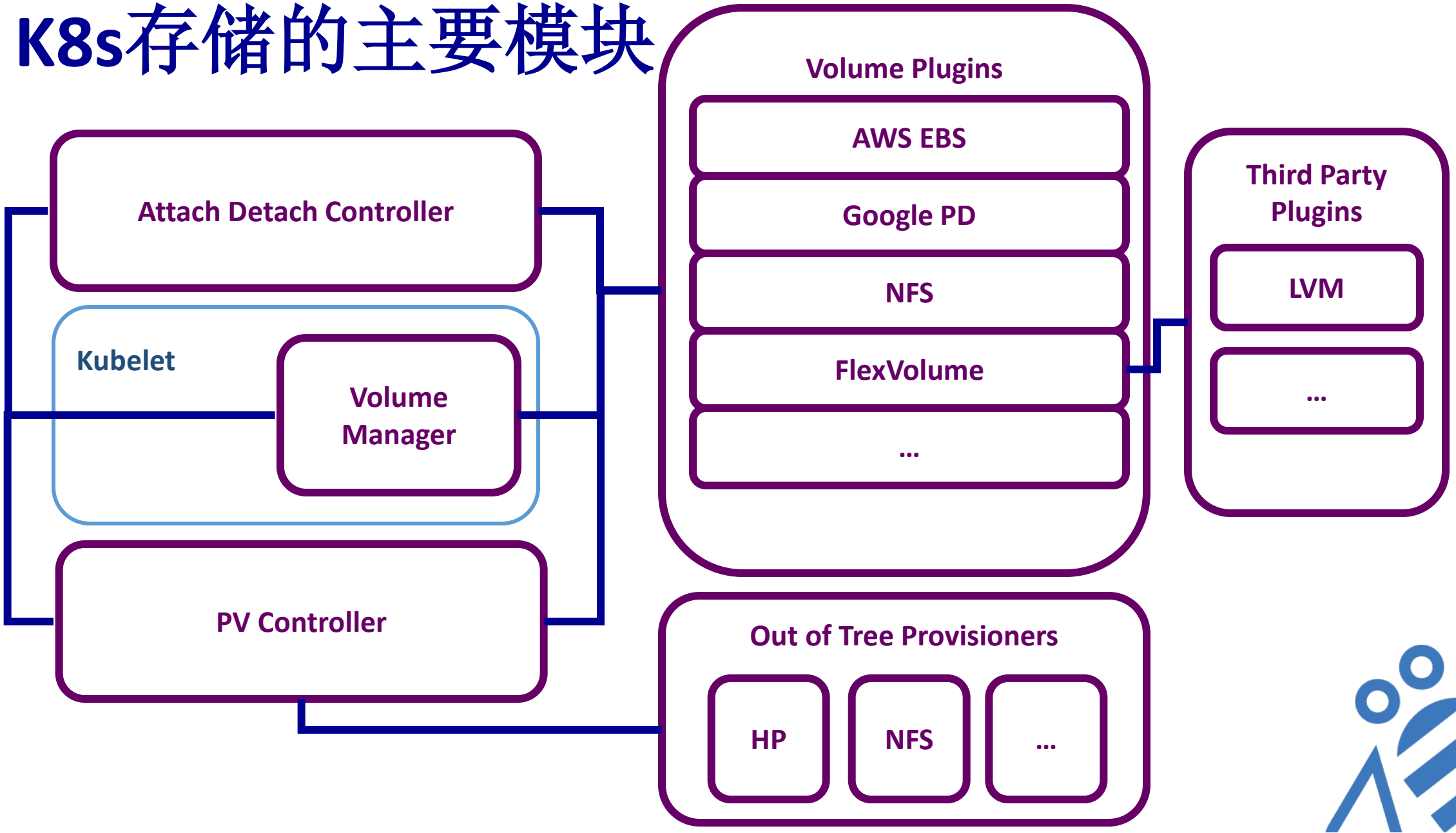
```
kubectl create –f nginx_pod.yaml

aws ec2 describe-volumes --volume-ids vol-xxxxxxxxxxxxx
{
   "Volumes": [
       {"Attachments": [
         {
           "VolumeId": "vol-xxxxxxxxxxxxxxxxx",
           "State": "attached”,
           "Device": "/dev/xvdba"
         }
       ]}
   ]
}
```

K8s存储的主要设计原则

- 遵循K8s整体架构
  - 声明式架构
- 易用性，尽可能多地兼容各种存储平台
  - 相比较于Docker Volume而言
  - 插件化
  - 兼容用户自定制插件
- 安全性
  - 数据安全性
  - 生命周期

K8s存储的主要模块


Volume Plugins
AWS EBS
Google PD
NFS
FlexVolume
...
Third Party Plugins
LVM
...
Attach Detach Controller
Kubelet
Volume Manager
PV Controller
Out of Tree Provisioners
HP
NFS
...

K8s存储的主要模块——Attach_Detach_Controller

- 负责将远程网络块存储设备挂载到某一个K8s节点的Controller
  - 两个存储结构
    - Actual State of World
    - Desired State of World
  - 三个线程
    - PopulateActualStateofWorld
    - PopulateDesiredStateofWorld
    - Reconcile
  - 与Volume Plugin的交互
    - Attach/Detach
    - Attachable Plugins

K8s存储的主要模块——PV Controller(1/6)

- Persistent Volume/Persistent Volume Claim
  - 从Storage Admin与用户的角度看PV与PVC
    - Admin创建和维护PV
    - 用户只需要使用PVC(size & access mode)
  - PVC与PV的绑定
  - 用户级别的逻辑对象，将Volume实现与Pod解耦
  - PVC与Volume
  - PV与Volume

K8s存储的主要模块——PV Controller(2/6)

- StorageClass
  - StorageClass将说明Volume由哪种Volume Provisioner创建、创建时参数以及从其他功能性 / 非功能性角度描述的后台volume的各种参数
    - Static Provisioning
    - Dynamic Provisioning
      - In-tree provisioner
      - Out-of-tree provisioner

K8s存储的主要模块——PV Controller(3/6)

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: myclaim
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 10Gi
  storageClassName: slow
```

```
kind: PersistentVolume
apiVersion: v1
metadata:
    name: pv0003
spec:
    capacity:
      storage: 5Gi
    accessModes:
      - ReadWriteOnce
    persistentVolumeReclaimPolicy: Recycle
    storageClassName: slow
    awsElasticBlockStore:
      volumeID: vol-xxxxxx
      fsType: ext4
```

```
kind: StorageClass
apiVersion: storage.k8s.io/v1
metadata:
  name: slow
provisioner: kubernetes.io/aws-ebs
parameters:
  parameters:
  type: io1
  zone: us-east-1d
  iopsPerGB: "10"
```

K8s存储的主要模块——PV Controller(4/6)

```
apiVersion: v1
kind: Pod
metadata:
    name: nginx
spec:
    containers:
    - image: nginx
      name: nginx-server
      volumeMounts:
      - mountPath: /var/nginx-data
        name: data-volume
    volumes:
    - name: data-volume
      awsElasticBlockStore:
        volumeID: vol-xxxxxxxxxxxxxxxxx
        fsType: ext4
```

```
apiVersion: v1
kind: Pod
metadata:
    name: nginx
spec:
    containers:
    - image: nginx
      name: nginx-server
      volumeMounts:
      - mountPath: /var/nginx-data
        name: data-volume
    volumes:
    - name: data-volume
      persistentVolumeClaim:
        claimName: myclaim
```

K8s存储的主要模块——PV Controller(5/6)

- PV Controller
  - 监控和管理集群中的PV和PVC
    - 实现PV/PVC生命周期管理
      - PVC: Pending, Bound, Lost
      - PV: Pending, Available, Bound, Released, Failed
    - 实现PV/PVC绑定
    - 实现Dynamic Provision

K8s存储的主要模块——PV Controller(6/6)

- Out-of-Tree Provisioner
  - 运行在K8s集群外，由用户自己实现
  - 使用时，在StorageClass中的provisioner字段注明provisioner名字即可
  - 需要监控K8s集群中etcd server中的PVC等数据
  - 需要实现init, provision, delete等接口
  - K8s孵化项目，社区实现了NFS等样例实现
  - https://github.com/kubernetes-incubator/external-storage

K8s存储的主要模块——Volume Manager(1/2)

- 运行在每个Kubelet上的核心模块
  - 用于协调attach/detach controller， PV controller和各个Volume Plugin
  - 最终实现将块设备从创建、设备挂载到挂载到K8s上指定目录的过程

K8s存储的主要模块——Volume Manager(1/2)

- K8s挂载卷的基本过程
  - 用户创建Pod包含一个PVC
  - Pod被分配到节点NodeA
  - Kubelet等待Volume Manager准备设备
  - PV controller调用相应Volume Plugin(in-tree或者out-of-tree)创建持久化卷并在系统中创建PV对象以及其与PVC的绑定(Provision)
  - Attach/Detach controller或者Volume Manager通过Volume Plugin实现块设备挂载(Attach)
  - Volume Manager等待设备挂载完成，将卷挂载到节点指定目录(mount)
    - /var/lib/kubelet/plugins/kubernetes.io/aws-ebs/mounts/vol-xxxxxxxxxxxxxxxxx
  - Kubelet在被告知设备准备好后启动Pod中的容器，利用Docker –v等参数将已经挂载到本地的卷映射到容器中(volume mapping)

K8s存储的主要模块——Volume Plugin(1/2)

- Volume Plugin的基本接口：
  - Init
  - Provision
  - Delete
  - Attach
  - Detach
  - Mount
  - Unmount

Provisioner Plugin（Provision、Delete）

Attachable Plugin（Attach、Detach）

K8s存储的主要模块——Volume Plugin(2/2)

- 持久化存储（网络）
  - Google Persistent Disk
  - AWS Elastic Block Store
  - Azure File Storage
  - Azure Data Disk
  - iSCSI
  - Flocker
  - NFS
  - vShpere
  - GlusterFS
  - Ceph File and RBD
  - Cinder
  - Quobyte Volume
  - FibreChannel
  - VMWare Photon PD
  - Portworx
  - Dell EMC ScaleIO
- 临时存储（本地）
  - Empty Dir(tmpfs)
  - K8s API
    - Secret
    - ConfigMap
    - Downward API
    - ProjectedVolume
- 其他
  - Flex Volume
  - Host Path
  - Local Persistent Storage

K8s存储的社区开发

- SIG-Storage
  - Slack Channel: https://kubernetes.slack.com/messages/sig-storage
  - Mailing list: https://groups.google.com/forum/#!forum/kubernetes-sig-storage
  - Sig-Storage Meeting: 每两周一次，周四16:00UTC
- 未来的主要方向
  - Resizing
  - Replication
  - Snapshot
  - CSI
  - …

一个简单的例子

在虚拟机中启动一个all-in-one K8s集群以及配置一个NFS服务器

通过NFS暴露一个共享目录

创建PV,PVC和Pod, Pod将挂载NFS所暴露的目录

谢谢大家