kubernetes笔记

笔记本：	新课程笔记		
创建时间：	2018/3/28 星期三 下午 2:46	**更新时间：**	2018/10/10 星期三 下午 4:28
作者：	306798658@qq.com		
URL：	https://kubernetes.io/		

官网 https://kubernetes.io/

Kubernetes也就是k8s

Kubernetes is an open-source system for automating deployment, scaling, and management of containerized applications.
Kubernetes是一个开源系统，它主要用来自动部署、扩容缩容和管理容器应用。

It groups containers that make up an application into logical units for easy management and discovery. Kubernetes builds upon 15 years of experience of running production workloads at Google, combined with best-of-breed ideas and practices from the community.
它将诸多应用的容器分为若干个逻辑单元以便于管理和发现。Kubernetes拥有着Google高负载生产环境的15年经验，并结合了社区的优秀思想和实践。

在kubernetes中，service是核心，我们并不需要太多关注kubernetes里面是怎么工作的，我们只需要关心它给我们提供什么service。

就像docker容器可以提供一个mysqld的服务、web服务等。

它需要拥有一个唯一的名字、有ip：port对外提供服务。

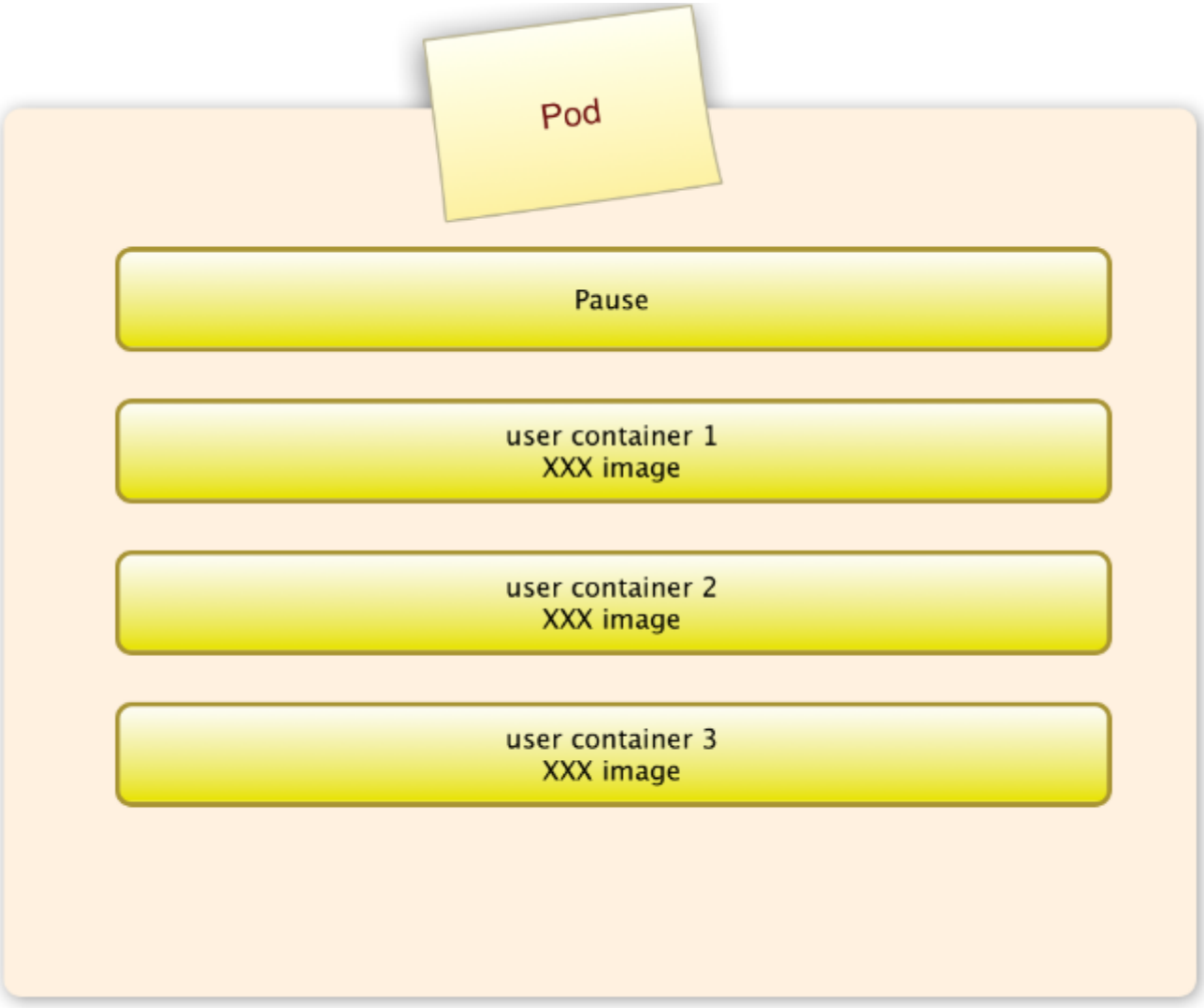
提供service的是容器，为了保证service的高可用，提供service的容器不能只有一个，需要一组。这一组容器我们把它叫做pod。

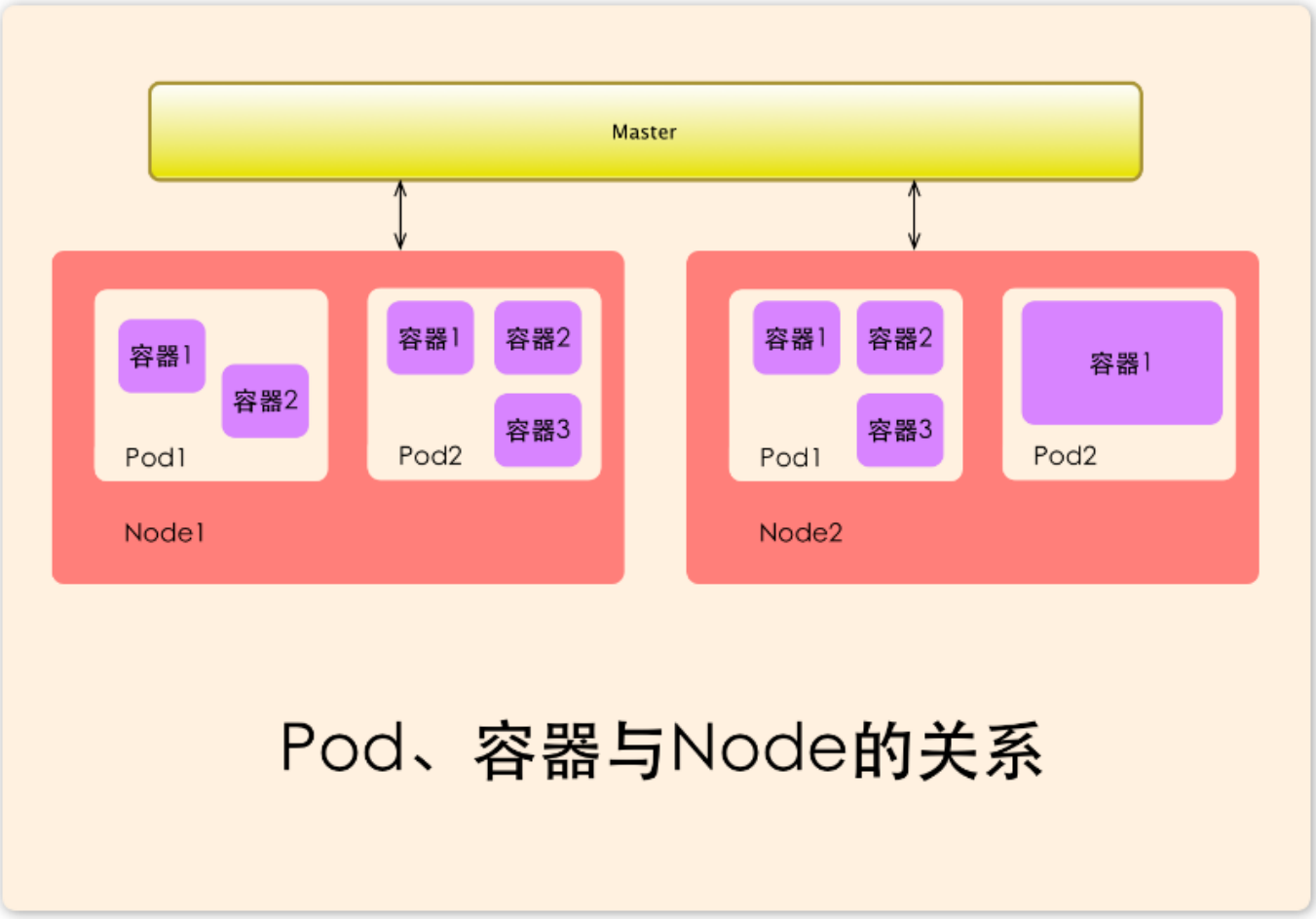
为了实现service和pod之间的关联，又有了标签（label）的概念，我们把功能相同的pod设定为同一个标签，比如，可以把所有提供mysql服务的pod贴上标签name = mysql，这样mysql service要作用于所有包含name = mysql标签的pod上。

pod运行在Node上，Node可以是一台物理机，也可以是虚拟机，通常一个Node上会运行几百个pod。每个pod里运行着一个特殊的容器，叫做Pause，其他容器叫做业务容器，业务容器共享Pause容器的网络栈和Volume挂载卷，因此同一个pod内的业务容器之间的通信和数据交换更为高效。

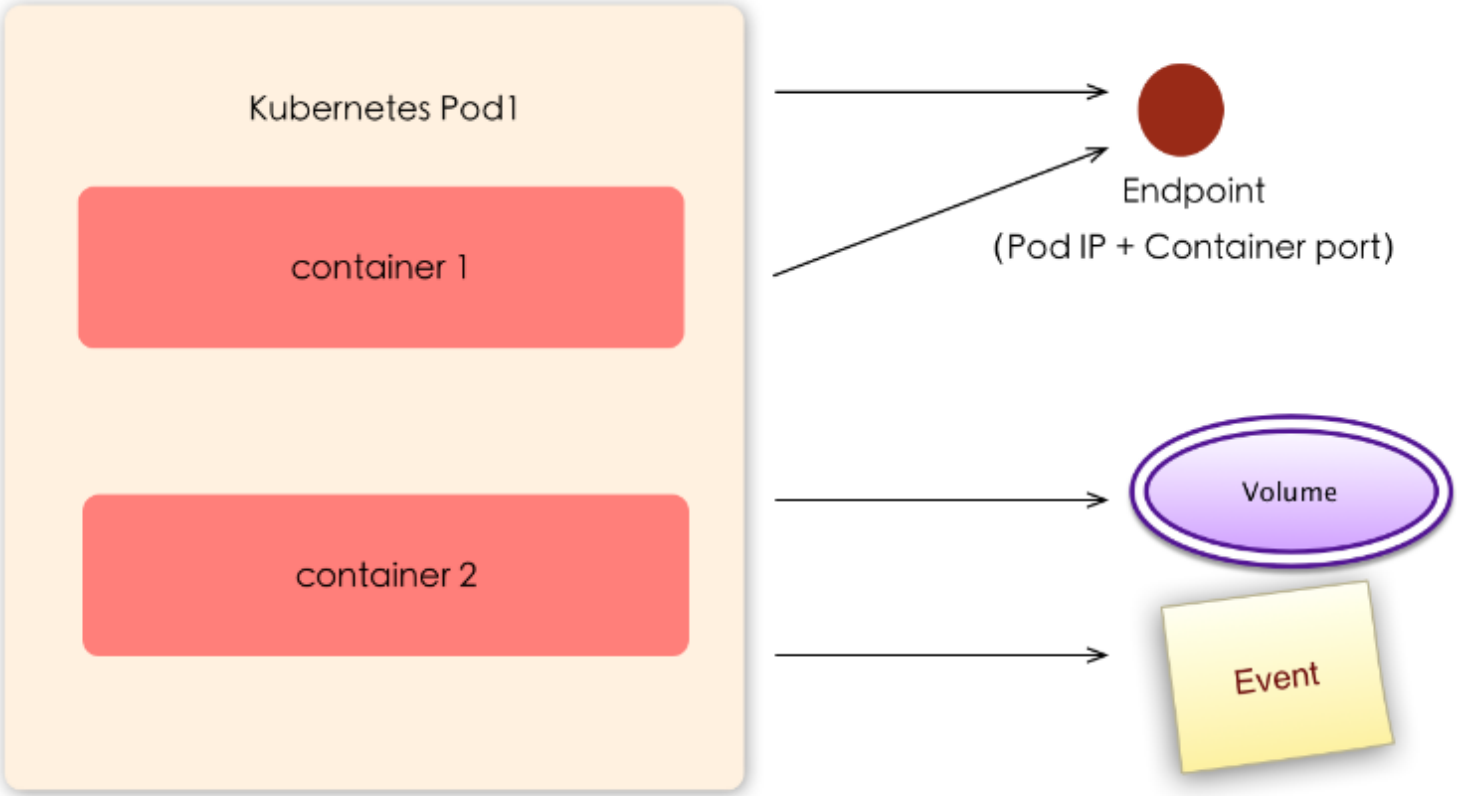
在集群管理方面，kubernetes将集群中的机器划分为一个master节点和一群工作节点Node，其中master上运行着kube-apiserver、kube-controller-manager、kube-scheduler，它们实现了资源管理、pod调度、弹性伸缩、安全控制、系统监控、纠错等功能。Node是工作节点，运行应用程序，提供服务。Node上的最小单元是pod，Node上运行着kubernetesd的kubelet、kube-proxy服务进程，它们负责pod的创建、启动、监控、重启、销毁，以及实现负载均衡。

通过一组图了解kubernetes各个元素的关系

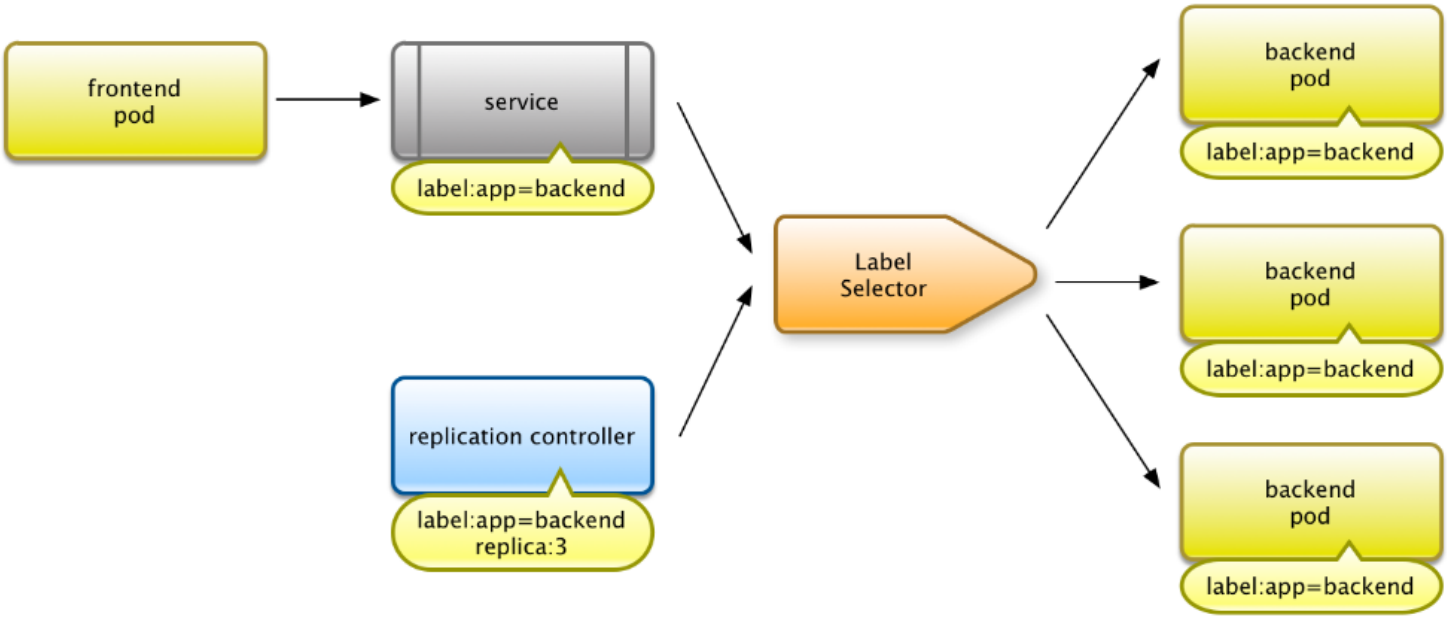

Pod
Pause
user container 1
XXX image
user container 2
XXX image
user container 3
XXX image



Pod、容器与Node的关系



Pod及周边对象



Pod、RC与Service的关系

扩容和升级需要一个关键的东西，Replication controller（RC），RC需要包含3个关键信息：
1）目标pod的定义
2）目标pod需要运行的副本数量（replicas）
3）要监控的目标pod的标签（Label）

工作过程：RC里定义好3个指标，kubernetes会根据RC定义的Label帅选出对应的pod，并实时监控其状态和数量，当实例数量少于定义的副本数（replicas），则会根据RC定义的pod模版来创建新的pod，然后将此pod调度到合适的Node上启动并运行。该过程完全自动化，无需人工干涉。

从一个例子开始：
webapp + mysql

安装kubernetes
准备一台centos7
1） 关闭firewalld 和 selinux

```
systemctl stop firewalld
systemctl disable firewalld
setenforce 0
```

2）安装etcd和kubernetes

```
yum install -y etcd kubernetes
```

3）修改配置文件

```
vi  /etc/sysconfig/docker
```

将--selinux-enabled 改为 --selinux-enabled=false --insecure-registry gcr.io

```
vi /etc/kubernetes/apiserver
```

把--admission_control参数中的ServiceAccount删除

4）准备工作

yum install python-rhsm-certificates
如果提示python-rhsm-certificates-1.19.10-1.el7_4.x86_64 被已安装的 subscription-manager-rhsm-certificates-1.20.11-1.el7.centos.x86_64 取代
wget http://mirror.centos.org/centos/7/os/x86_64/Packages/python-rhsm-certificates-1.19.10-1.el7_4.x86_64.rpm
rpm2cpio python-rhsm-certificates-1.19.10-1.el7_4.x86_64.rpm |cpio -iv --to-stdout ./etc/rhsm/ca/redhat-uep.pem > /etc/rhsm/ca/redhat-uep.pem

配置docker加速器

```
vi /etc/docker/daemon.json//加入如下内容
{
  "registry-mirrors": ["https://dhq9bx4f.mirror.aliyuncs.com"]
}
```

5）按顺序启动所有服务

```
for s in etcd docker kube-apiserver  kube-controller-manager kube-scheduler  kubelet  kube-proxy
do
    systemctl start $s
done
```

6）创建一个rc文件

```
vim mysql-rc.yaml  #内容如下
apiVersion: v1
kind: ReplicationController                    #副本控制器RC
metadata:
  name: mysql                                  #RC的名称，全局唯一
spec:
  replicas: 1                                  #Pod副本的期待数量
  selector:
    app: mysql                                 #符合目标的Pod拥有此标签
  template:                                    #根据此模板创建Pod的副本（实例）
    metadata:
      labels:
        app: mysql                             #Pod副本拥有的标签，对应RC的Selector
    spec:
      containers:                              #Pod内容器的定义部分
      - name: mysql                            #容器的名称
        image: mysql:5.6                           #容器对应的Docker image
        ports:
        - containerPort: 3306                  #容器应用监听的端口号
        env:                                   #注入容器内的环境变量
        - name: MYSQL_ROOT_PASSWORD
          value: "123456"
```

```
docker pull registry.access.redhat.com/rhel7/pod-infrastructure:latest
docker pull mysql:5.6

kubectl  create -f mysql-rc.yaml
kubectl get rc
kubectl get pods
```

7）创建一个svc文件

```
vim mysql-svc.yaml
apiVersion: v1
kind: Service
metadata:
  name: mysql
spec:
  ports:
    - port: 3306
  selector:
    app: mysql

kubectl create -f mysql-svc.yaml
kubectl get svc
```

8）创建web rc文件

```
vim web-rc.yaml
kind: ReplicationController
metadata:
  name: myweb
spec:
  replicas: 1
  selector:
    app: myweb
  template:
    metadata:
      labels:
        app: myweb
    spec:
      containers:
        - name: myweb
          image: kubeguide/tomcat-app:v1
          ports:
          - containerPort: 8080
          env:
```

```
          - name: MYSQL_SERVICE_HOST
            value: 'mysql_service_ip'  #这里的IP需要通过kubect get svc 查看mysql的cluster ip
          - name: MYSQL_SERVICE_PORT
            value: '3306'

kubectl create -f web-rc.yaml
```

9）创建web svc文件

```
vim web-svc.yamly
kind: Service
metadata:
  name: myweb
spec:
  type: NodePort
  ports:
    - port: 8080
      nodePort: 30001
  selector:
    app: myweb

kubectl create -f web-svc.yaml
```

10）访问
iptables -P FORWARD ACCEPT
curl 本机ip:30001/demo/ 或浏览器

问题
https://blog.csdn.net/gezilan/article/details/80011905
https://www.cnblogs.com/neutronman/p/8047547.html
https://blog.csdn.net/d7185540/article/details/80868816