实验 07-Kubernetes Cluster

一、实验目的

1、掌握容器化云数据中心的方案设计方法；

2、掌握 Kubernetes Cluster 的部署和配置；

3、掌握可视化管理工具 Kuboard 的安装和使用。

二、实验学时

2 学时

三、实验类型

设计性

四、实验任务

1、完成容器化云数据中心的方案设计；

2、完成 Kubernetes Cluster 的部署和配置；

3、完成可视化管理工具 Kuboard 的安装。

五、实验环境

1、硬件

本实验基于实验教学中心网络运维实验室服务器集群开展，每个实验小组分配集群中的 1 台物理服务器作为实验基础平台，提供云计算资源。每个人配备计算机 1 台。（学生可根据自身情况使用个人计算机）。

2、软件

Windows 操作系统，或 MacOS 操作系统。

安装最新版本的浏览器，建议使用 Edge、Chrome 等。

3、网络

计算机使用无线网络接入局域网，能够访问实验教学中心网络运维实验室服务器集群，并支持对互联网的访问。

4、工具

无

六、实验内容步骤

1、云数据中心方案设计

（1）应用场景

建设一个容器化云数据中心。

在容器化云数据中心上创建 5 个用户，每个用户分配：

（1）独立的命名空间实现资源隔离

（2）独立的 NFS 共享存储

（3）独立内部地址段用于容器服务

（4）对外服务的 10 个 IP 地址

容器化云数据中心应用规划如表 7-1 所示。

表 7-1 容器化云数据中心应用规划表

序号	用户名	分配命名空间	分配存储	内部地址	
1	user01	namespace01	cloud-k8s-nfs-1	192.168.1.0/24	172.16
2	user02	namespace02	cloud-k8s-nfs-2	192.168.2.0/24	172.16
3	user03	namespace03	cloud-k8s-nfs-3	192.168.3.0/24	172.16
4	user04	namespace04	cloud-k8s-nfs-4	192.168.4.0/24	172.16
5	user05	namespace05	cloud-k8s-nfs-5	192.168.5.0/24	172.16

（2）节点规划

容器化云数据中心采用 Kubernetes 技术栈建设，由 5 个节点组成，主节点 1 个，工作节点 4 个。

节点使用虚拟机，配置信息如表 7-2 所示。

表 7-2 集群节点配置信息

序号	虚拟机配置	操作系统配置
1	虚拟机名称：Cloud-K8s-Master CPU：2 核 内存：4GB 硬盘：50GB 网卡：Class-Cloud-VM-Network	主机名：Cloud-K8s-Master 操作系统：openEuler 24.03 LTS S IP 地址：172.16.125.101 子网掩码：255.255.255.0 网关：172.16.125.1 DNS：172.16.125.3
2	虚拟机名称：Cloud-K8s-Worker-1 CPU：2 核 内存：4GB 硬盘：50GB 网卡：Class-Cloud-VM-Network	主机名：Cloud-K8s-Worker-1 操作系统：openEuler 24.03 LTS S IP 地址：172.16.125.102 子网掩码：255.255.255.0 网关：172.16.125.1 DNS：172.16.125.3
3	虚拟机名称：Cloud-K8s-Worker-2 CPU：2 核 内存：4GB 硬盘：50GB 网卡：Class-Cloud-VM-Network	主机名：Cloud-K8s-Worker-2 操作系统：openEuler 24.03 LTS S IP 地址：172.16.125.103 子网掩码：255.255.255.0 网关：172.16.125.1 DNS：172.16.125.3
4	虚拟机名称：Cloud-K8s-Worker-3 CPU：2 核 内存：4GB 硬盘：50GB 网卡：Class-Cloud-VM-Network	主机名：Cloud-K8s-Worker-3 操作系统：openEuler 24.03 LTS S IP 地址：172.16.125.104 子网掩码：255.255.255.0 网关：172.16.125.1 DNS：172.16.125.3
5	虚拟机名称：Cloud-K8s-Worker-4 CPU：2 核 内存：4GB 硬盘：50GB 网卡：Class-Cloud-VM-Network	主机名：Cloud-K8s-Worker-4 操作系统：openEuler 24.03 LTS S IP 地址：172.16.125.105 子网掩码：255.255.255.0 网关：172.16.125.1 DNS：172.16.125.3

（3）网络规划

（1）节点网络

- 主节点：`172.16.125.101`
- 工作节点：`172.16.125.102`、`172.16.125.103`、`172.16.125.104`、`172.16.125.105`

（2）集群内部网络

Pod 网络是整个 Kubernetes 集群的 Pod 网络范围，所有 Pod 的 IP 地址都从该范围内分配。

Pod 网络 CIDR：`192.168.0.0/16`

Service 网络是 Kubernetes 集群中 Service 的虚拟 IP 地址范围，用于集群内部的服务发现和负载均衡。

Service 网络 CIDR：`10.96.0.0/12`

2、配置系统环境

步骤 1：在节点 Cloud-K8s-Master 上配置 SElinux、防火墙和 Swap 分区。

Shell

```
# 关闭防火墙
systemctl stop firewalld

# 禁用防火墙开机自启动
systemctl disable firewalld

# 临时关闭SElinux
setenforce 0

# 永久关闭SElinux
sed -i 's/enforcing/disabled/' /etc/selinux/config

# 永久关闭 Swap 分区
sed -ri 's/.*swap.*/#&/' /etc/fstab
```

步骤 2：设置主机时间同步。

```shell
# 设置节点执行时间同步：
yum install ntpdate -y
ntpdate 172.16.125.3
```

步骤 3：设置主机名并配置 hosts 文件。

```shell
# 设置主机名 后面的4台主机也要设置
hostnamectl set-hostname k8s-master  //注意每台主机名称不同

# 在节点上添加 hosts 文件的配置
cat >> /etc/hosts << EOF
172.16.125.101 k8s-master
172.16.125.102 k8s-worker1
172.16.125.103 k8s-worker2
172.16.125.104 k8s-worker3
172.16.125.105 k8s-worker4
EOF
```

步骤 4：配置网桥过滤和地址转发。

```shell
# 为所有节点添加网桥过滤和地址转发功能
cat > /etc/sysctl.d/k8s.conf << EOF
net.bridge.bridge-nf-call-ip6tables = 1
net.bridge.bridge-nf-call-iptables = 1
EOF

# 应用新的配置
sysctl --system

# 重启服务器
reboot
```

参照节点 Cloud-K8s-Master 的系统环境配置，完成 Cloud-K8s-Worker-1、Cloud-K8s-Worker-2、Cloud-K8s-Worker-3、Cloud-K8s-Worker-4 系统环境配置。

3、安装 Kubernetes 基础软件

步骤 1：安装 Docker。

Shell

```
#配置安装源
yum-config-manager --add-repo http://mirrors.aliyun.com/docker-ce/linux/centos/docker-ce.repo
sed -i 's/\$releasever/8/g' /etc/yum.repos.d/docker-ce.repo

# 安装Docker
yum install -y docker-ce docker-ce-cli containerd.io
```

步骤 2：配置镜像加速。

Shell

```
# 配置镜像加速和cgroup驱动
cat > /etc/docker/daemon.json <<EOF
{
  "registry-mirrors": [
    "https://registry.cn-hangzhou.aliyuncs.com",
    "https://hub.xdark.top",
    "https://hub.littlediary.cn",
    "https://dockerpull.org",
    "https://hub.crdz.gq",
    "https://docker.1panel.live",
    "https://docker.mirrors.ustc.edu.cn",
    "https://docker.m.daocloud.io",
    "https://noohub.ru",
    "https://huecker.io",
    "https://dockerhub.timeweb.cloud",
    "https://docker.1panel.dev",
    "https://docker.unsee.tech",
    "https://docker.1panel.live"
],
  "exec-opts": ["native.cgroupdriver=systemd"],
  "log-driver": "json-file",
  "log-opts": {
    "max-size": "100m"
  }
}
EOF

#启动docker并设置开机自启动
systemctl start docker
systemctl enable docker
```

步骤 3：安装 cri-dockerd。

Shell

```
# 安装cri-dockerd，让 Docker 作为 K8s 容器引擎
wget https://github.com/Mirantis/cri-dockerd/releases/download/v0.3.12/cri-dockerd-0.3.12-3.el7.x86_64.rpm
rpm -ivh cri-dockerd-0.3.12-3.el7.x86_64.rpm

# 修改配置文件（/usr/lib/systemd/system/cri-docker.service）
#在 “ExecStart=/usr/bin/cri-dockerd --container-runtime-endpoint fd://” 这一行增加
 “--pod-infra-container-image=registry.aliyuncs.com/google_containers/pause:3.9”

vi /usr/lib/systemd/system/cri-docker.service
-------------------cri-docker.service--------------
ExecStart=/usr/bin/cri-dockerd --container-runtime-endpoint fd:// --pod-infra-container-image=registry.aliyuncs.com/google_containers/pause:3.9
----------------------------------------------------

#加载配置并开启服务
systemctl daemon-reload
systemctl enable cri-docker && systemctl start cri-docker
```

步骤 4：安装 kubeadm、kubelet 和 kubectl。

Shell

```
#安装kubeadm、kubelet 和 kubectl
#添加阿里云的镜像库
cat > /etc/yum.repos.d/kubernetes.repo << EOF
[kubernetes]
name=Kubernetes
baseurl=https://mirrors.aliyun.com/kubernetes/yum/repos/kubernetes-el7-x86_64
enabled=1
gpgcheck=0
repo_gpgcheck=0
gpgkey=https://mirrors.aliyun.com/kubernetes/yum/doc/yum-key.gpg https://mirrors.aliyun.com/kubernetes/yum/doc/rpm-package-key.gpg
EOF

#安装kubeadm、kubelet 和 kubectl
yum install -y kubelet-1.28.0 kubeadm-1.28.0 kubectl-1.28.0

#设置开机启动 kubelet 服务
systemctl enable kubelet
```

4、初始化 Kubernetes（Master 节点）

Shell

```
#在 Master 节点上执行以下初始化命令
kubeadm init \
  --apiserver-advertise-address=172.16.125.101 \
  --image-repository registry.aliyuncs.com/google_containers \
  --kubernetes-version v1.28.0 \
  --service-cidr=10.96.0.0/12 \
  --pod-network-cidr=192.168.0.0/16 \
  --cri-socket=unix:///var/run/cri-dockerd.sock \
  --ignore-preflight-errors=all

#apiserver-advertise-address：集群广播地址，用 master 节点的内网 IP
#image-repository：由于默认拉取镜像地址 k8s.gcr.io 国内无法访问，这里指定阿里云镜像仓库地址
#kubernetes-version： K8s 版本，与上面安装的软件版本一致
#service-cidr：集群 Service 网段
#pod-network-cidr：集群 Pod 网段
#cri-socket：指定 cri-socket 接口，使用 unix:///var/run/cri-dockerd.sock


//初始化完成后出现提示内容：
Your Kubernetes control-plane has initialized successfully!

To start using your cluster, you need to run the following as a regular user:

  mkdir -p $HOME/.kube
  sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
  sudo chown $(id -u):$(id -g) $HOME/.kube/config

Alternatively, if you are the root user, you can run:

  export KUBECONFIG=/etc/kubernetes/admin.conf

You should now deploy a pod network to the cluster.
Run "kubectl apply -f [podnetwork].yaml" with one of the options listed at:
  https://kubernetes.io/docs/concepts/cluster-administration/addons/

Then you can join any number of worker nodes by running the following on each as root:

kubeadm join 172.16.125.101:6443 --token wqhlp8.pbepvo8rkynn3q5z \
```

```
39         --discovery-token-ca-cert-hash sha256:f17c5cb118e2c8834eab43a96
   a83b29f033b7ca8dec49f9a5f52f6b8c627b57d
```

💡 提醒：

在后续集群的配置中需要用到初始化完成后出现提示内容，注意保存。

在 Master 节点设置 kubectl 工具的管理员权限。

Shell

```
1 #执行初始化完成后出现提示内容：
2 mkdir -p $HOME/.kube
3 sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
4 sudo chown $(id -u):$(id -g) $HOME/.kube/config
```

5、创建 Kubernetes 集群

在 Worker 节点上执行初始化完成返回的 kubeadm join 命令，为集群添加工作节点。

Shell

```
1 kubeadm join 172.16.125.101:6443 --token wqhlp8.pbepvo8rkynn3q5z \
2 --discovery-token-ca-cert-hash sha256:f17c5cb118e2c8834eab43a96a83b29f0
   33b7ca8dec49f9a5f52f6b8c627b57d
3 --cri-socket=unix:///var/run/cri-dockerd.sock
```

📌 命令中 token 有效期为 24 小时，当 token 过期之后，执行 “kubeadm join” 命令就会报错。这时可以直接在 Master 节点上使用以下命令生成新的 token，然后再使用 “kubeadm join” 命令加入节点。

创建一个新的 join token：

kubeadm token create --print-join-command

此时查看集群节点状态，各节点均加入集群但状态都是 NotReady，集群的内部网络还没有正常运作，需要安装网络插件。

Shell

```
[root@k8s-master ~]# kubectl get node
NAME          STATUS     ROLES           AGE     VERSION
k8s-master    NotReady   control-plane   20m     v1.28.0
k8s-worker1   NotReady   <none>          3m3s    v1.28.0
k8s-worker2   NotReady   <none>          2m30s   v1.28.0
k8s-worker3   NotReady   <none>          116s    v1.28.0
k8s-worker4   NotReady   <none>          86s     v1.28.0
```

6、安装网络插件 Calico（Master 节点）

Shell

```
# 下载 Calico 插件部署文件
wget https://docs.projectcalico.org/manifests/calico.yaml

# 修改 “calico.yaml” 文件中的 “CALICO_IPV4POOL_CIDR” 参数，需要与前面 “kubeadm init”
 命令中的 “-pod-network-cidr” 参数一样（192.168.0.0/16）
vi calico.yaml

#可以输入 “/CALICO_IPV4POOL_CIDR” 来快速定位到参数位置
------------calico.yaml-------------
- name: CALICO_IPV4POOL_CIDR
  value: "192.168.0.0/16"
-----------------------------------

# 将 Calico 插件部署到集群里
kubectl apply -f calico.yaml

#等几分钟后查看节点状态
 aster ~]# kubectl get node //状态STATUS是Ready说明集群创建成功
NAME          STATUS   ROLES           AGE   VERSION
k8s-master    Ready    control-plane   68m   v1.28.0
k8s-worker1   Ready    <none>          51m   v1.28.0
k8s-worker2   Ready    <none>          50m   v1.28.0
k8s-worker3   Ready    <none>          50m   v1.28.0
k8s-worker4   Ready    <none>          49m   v1.28.0
```

7、安装 Kuboard

Shell

```
#在 master 节点添加 k8s.kuboard.cn/role=etcd 的标签
kubectl label nodes k8s-master k8s.kuboard.cn/role=etcd
kubectl label nodes k8s-worker1 k8s.kuboard.cn/role=etcd
kubectl label nodes k8s-worker2 k8s.kuboard.cn/role=etcd
kubectl label nodes k8s-worker3 k8s.kuboard.cn/role=etcd
kubectl label nodes k8s-worker4 k8s.kuboard.cn/role=etcd

#下载 kuboard 部署清单
wget https://addons.kuboard.cn/kuboard/kuboard-v3-swr.yaml

#部署 kuboard
kubectl apply -f kuboard-v3-swr.yaml

#查看 pod 信息
kubectl get pods -n kuboard
```

部署完成后浏览器访问：http://172.16.125.101:30080，登录后如图 7-1 所示。

> 提醒：
>
> Kuboard 部署后的初始用户名：admin，密码：Kuboard123。

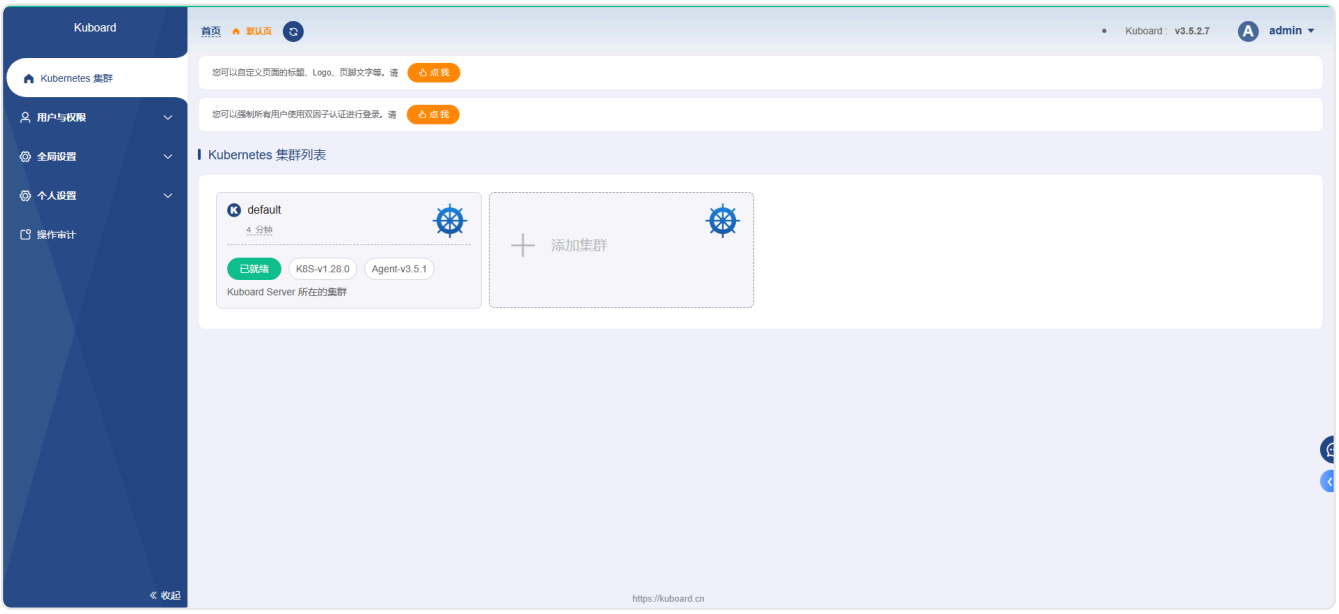

图7-1 Kuboard初始界面

添加本地集群，如图 7-2 所示。

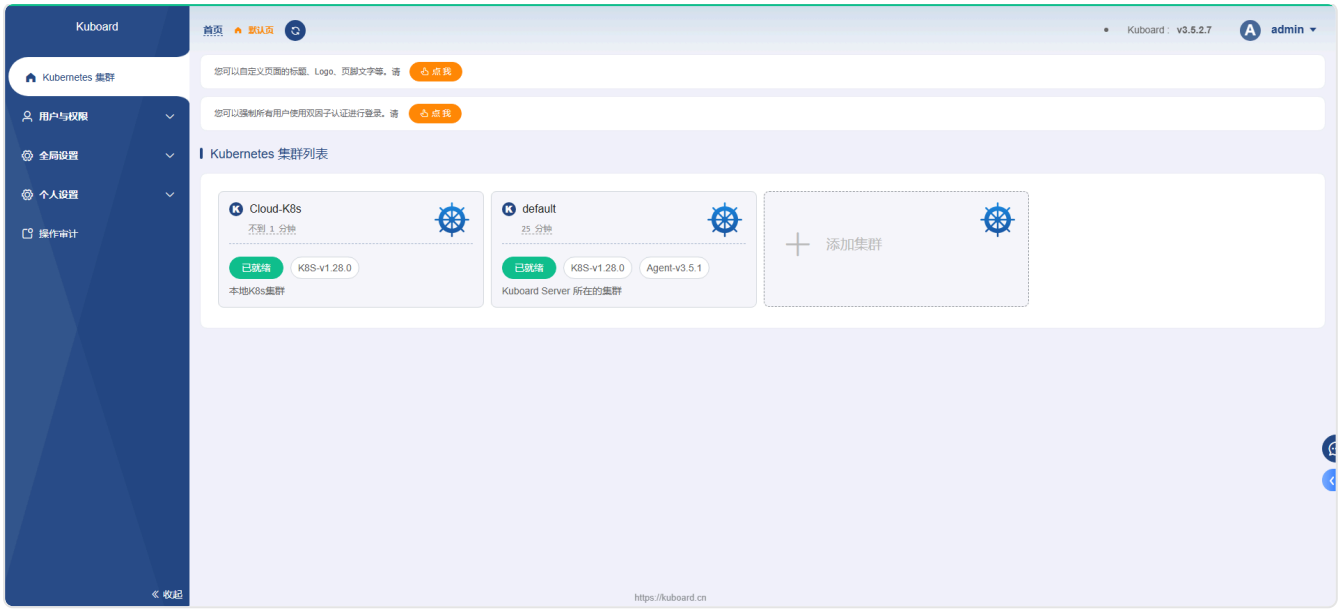

图 7-2 添加本地集群

七、实验讲解

本实验配套讲解视频，访问课程学习平台。

八、实验考核

实验考核为【实验随堂查】。

实验随堂查：每个实验设置 3-5 考核点，学生现场进行演示和汇报讲解。

1、考核点

考核点 1：完成 K8s 集群的搭建，集群状态为 Ready。（40 分）

考核点 2：完成可视化界面 Kuboard 的安装。（30 分）

考核点 3：在 Kuboard 中添加本地 K8s 集群。（30 分）

2、考核方式

以实验小组为单位进行考核，每个小组由 1 位同学进行实验成果汇报，小组其他成员回答教师提问。根据汇报和答疑情况，对小组成员进行逐一打分。

由教师进行评分。