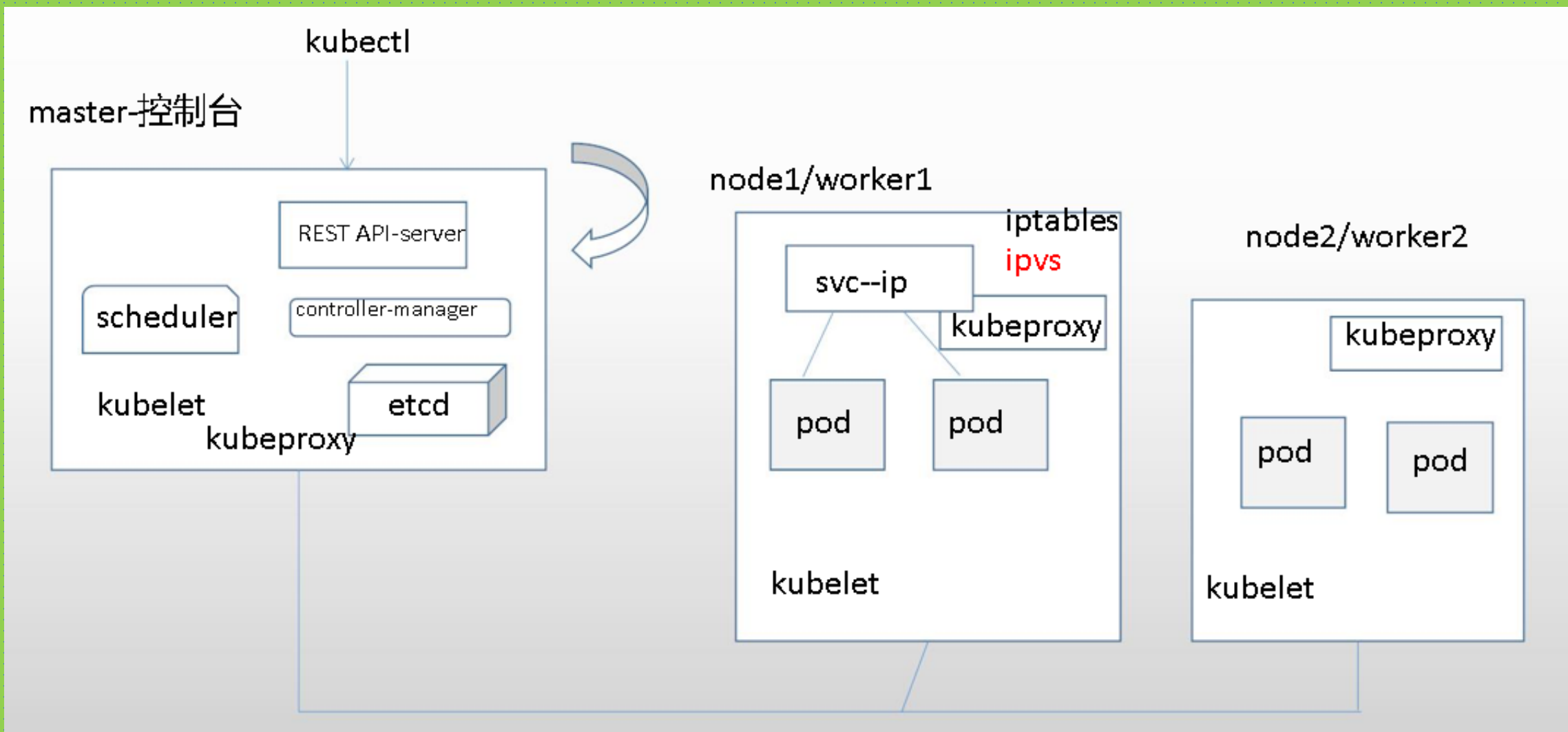




CKA-kubernetes安装



kubernetes框架介绍





概念介绍

- **kubectl**: 客户端命令行工具，将接受的命令格式化后发送给kube-apiserver，作为整个系统的操作入口。
- **kube-apiserver**: 作为整个系统的控制入口，以REST API服务提供接口。
- **kube-controller-manager**: 用来执行整个系统中的后台任务，包括节点状态状况、Pod个数、Pods和Service的关联等。
- **kube-scheduler**: 负责节点资源管理，接受来自kube-apiserver创建Pods任务，并分配到某个节点。
- **etcd**: 负责节点间的服务发现和配置共享。
- **kube-proxy**: 运行在每个计算节点上，负责Pod网络代理。定时从etcd获取到service信息来做相应的策略。
- **kubelet**: 运行在每个计算节点上，作为agent，接受分配该节点的Pods任务及管理容器，周期性获取容器状态，反馈给kube-apiserver。



kubeadmin的安装方式

--在master和node上执行

配置系统

关闭防火墙, selinux, 配置/etc/hosts, 关闭swap, 配置yum

安装docker

yum install docker -y , 启动docker并设置开机自动启动, 导入镜像

设置相关属性

```
cat <<EOF > /etc/sysctl.d/k8s.conf
net.bridge.bridge-nf-call-ip6tables = 1
net.bridge.bridge-nf-call-iptables = 1
net.ipv4.ip_forward = 1
EOF
```

```
cat > /etc/docker/daemon.json <<EOF
{
  "registry-mirrors": ["https://i1pfdcu7.mirror.aliyuncs.com"]
}
EOF
```

安装kubernetes相关软件包

```
yum install -y kubelet-1.15.3-0 kubeadm-1.15.3-0 kubectl-1.15.3-0 --disableexcludes=kubernetes
yum install -y kubelet-1.18.1-0 kubeadm-1.18.1-0 kubectl-1.18.1-0 --disableexcludes=kubernetes
```

注意, 需要指定版本, 否则安装是最新版的

启动服务

```
systemctl restart kubelet ; systemctl enable kubelet
```



kubeadmin的安装方式

--在maser上执行

```
kubeadm init --kubernetes-version=v1.18.1 --pod-network-cidr=10.244.0.0/16
```

```
kubeadm init --image-repository registry.aliyuncs.com/google_containers --kubernetes-version=v1.18.1 --pod-network-cidr=10.244.0.0/16
```

```
kubeadm init --config kubeadm-config.yaml
```

安装flannel网络

```
kubectl apply -f https://raw.githubusercontent.com/coreos/flannel/v0.10.0/Documentation/kube-flannel.yml
```

```
kubectl apply -f https://docs.projectcalico.org/v3.3/getting-started/kubernetes/installation/hosted/rbac-kdd.yaml
```

```
kubectl apply -f https://docs.projectcalico.org/v3.3/getting-started/kubernetes/installation/hosted/kubernetes-datastore/calico-networking/1.7/calico.yaml
```

```
sed -i 's/192.168.0.0/10.244.0.0/g' calico_v3.10.yaml
```

配置KUBECONFIG变量

```
#echo "export KUBECONFIG=/etc/kubernetes/admin.conf" >> /etc/profile
```

```
source /etc/profile
```

kubeadmin的安装方式



把node加入到集群

`kubeadm join IP:6443 --token TOKEN` 这个命令上面有提示

如果后期忘记了, 可以通过`kubeadm token create --print-join-command`查看
重新部署环境 `kubeadm reset`

设置可以用tab补齐键 `vim /etc/profile`

`source <(kubectl completion bash)`

`source /etc/profile`

前提: `bash-completion.noarch` 必须要安装才行

vim的设置

`set paste`

`kubectl edit cm -n kube-system`

删除daemonset pod



常见命令

`kubectl cluster-info`

`kubectl version`

`kubectl api-versions`

`kubectl config view`

删除节点

`kubectl drain node1.example.com --delete-local-data --force --ignore-daemonsets`

`kubectl delete node node1.example.com`



设置Heapster

`docker pull registry.cn-hangzhou.aliyuncs.com/google_containers/heapster-amd64:v1.5.4`

运行heapster.yaml

运行heapster-mod.yaml

`kubectl top node`

`kubectl top pod`

```
# kubectl top nodes
Error from server (ServiceUnavailable): the server is currently unable to handle the r
p:heapster:)
```

```
# kubectl top nodes
error: metrics not available yet
```

如果出现如上错误，请在所有节点上开启转发功能
`echo 1 > /proc/sys/net/ipv4/ip_forward`



设置metric server

```
curl -Ls https://api.github.com/repos/kubernetes-sigs/metrics-server/tarball/v0.3.6 -o metrics-server-v0.3.6.tar.gz
```

修改metrics-server-deployment.yaml

```
docker pull mirrorgooglecontainers/metrics-server-amd64:v0.3.6
```

containers:

```
- name: metrics-server
  image: k8s.gcr.io/metrics-server-amd64:v0.3.6
  #imagePullPolicy: Always
  imagePullPolicy: IfNotPresent
  command:
  - /metrics-server
  - --kubelet-insecure-tls
  #- --kubelet-preferred-address-types=InternalIP
```

如果遇到dns没法解析

```
kubectl edit cm coredns -n kube-system
```

```
apiVersion: v1
data:
  Corefile: |
    .:53 {
      errors
      health {
        lameduck 5s
      }
      hosts {
        192.168.26.181 vms181.rhce.cc vms181
        192.168.26.182 vms182.rhce.cc vms182
        192.168.26.183 vms183.rhce.cc vms183
      }
      ready
      kubernetes cluster.local in-addr.arpa ip6.arpa {
```



了解namespace

不同的namespace之间互相隔离

```
kubectl config get-contexts
```

```
kubectl config set-context 集群名 --namespace=命名空间
```

```
kubectl config set-context --current --namespace=命名空间
```



手动安装--master(仅了解)

目的是，让大家感受到kubernetes的各个组件之间的关联

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

```
vim /etc/etcd/etcd.conf
```

```
ETCD_LISTEN_CLIENT_URLS="http://0.0.0.0:2379"
```

```
ETCD_ADVERTISE_CLIENT_URLS="http://0.0.0.0:2379"
```

```
systemctl restart etcd ; systemctl enable etcd
```

```
etcdctl -C http://192.168.26.31:2379 set /atomic.io/network/config '{"Network":"172.17.0.0/16"}
```

```
etcdctl ls /atomic.io/network/config
```

```
etcdctl get /atomic.io/network/config
```



手动安装--master

/etc/kubernetes/config

KUBE_MASTER="--master=http://192.168.26.31:8080"

/etc/kubernetes/apiserver

KUBE_API_ADDRESS="--insecure-bind-address=0.0.0.0"

KUBE_API_PORT="--port=8080"

KUBELET_PORT="--kubelet-port=10250"

/etc/kubernetes/controller-manager

KUBELET_ADDRESSES="--machines=192.168.26.31,192.168.26.32"



手动安装--master

```
KUBELET_ADDRESS="--address=0.0.0.0"
```

```
KUBELET_PORT="--port=10250"
```

```
KUBELET_HOSTNAME="--hostname-override=192.168.26.31"
```

```
KUBELET_API_SERVER="--api-servers=http://192.168.26.31:8080"
```

```
systemctl restart kube-apiserver.service kube-controller-  
manager.service kube-scheduler.service kubelet.service
```



手动安装--node

```
yum install kubernetes-node flannel -y
```

```
/etc/kubernetes/config
```

```
KUBE_MASTER="--master=http://192.168.26.31:8080"
```

```
/etc/kubernetes/kubelet
```

```
KUBELET_ADDRESS="--address=0.0.0.0"
```

```
KUBELET_PORT="--port=10250"
```

```
KUBELET_HOSTNAME="--hostname-override=192.168.26.32"
```

```
KUBELET_API_SERVER="--api-servers=http://192.168.26.31:8080"
```

```
/etc/sysconfig/flanneld
```

```
FLANNEL_ETCD_ENDPOINTS="http://192.168.26.31:2379"
```

```
systemctl start kubelet kube-proxy flanneld
```

多集群切换



apiVersion: v1

clusters:

- cluster:

1111111111

server: https://192.168.26.10:6443

name: kubernetes1

- cluster:

2222222222

server: https://192.168.26.21:6443

name: kubernetes2

contexts:

- context:

cluster: kubernetes1

user: kubernetes-admin1

name: kubernetes1

- context:

cluster: kubernetes2

user: kubernetes-admin2

name: kubernetes2

current-context: kubernetes1

kind: Config

preferences: {}

users:

- name: kubernetes-admin1

user:

1111

- name: kubernetes-admin2

user:

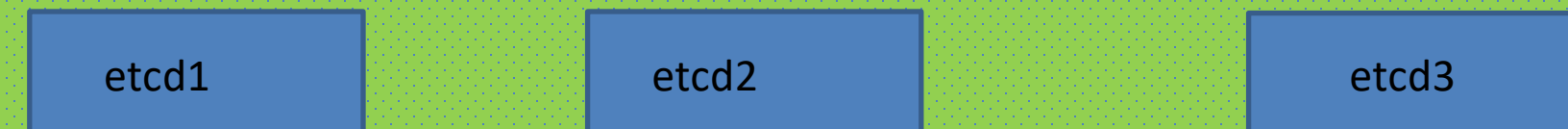
222

kubectl config get-contexts

kubectl config use-context kubernetes1



etcd流程



ETCD集群是一个分布式系统,使用Raft协议来维护集群内各个节点状态的一致性。
主机状态 Leader, Follower, Candidate

当集群初始化时候, 每个节点都是Follower角色
通过心跳与其他节点同步数据

当Follower在一定时间内没有收到来自主节点的心跳, 会将自己角色改变为Candidate, 并发起一次选主投票

配置etcd集群, 建议尽可能是奇数个节点, 而不要偶数个节点



单节点ETCD

```
yum install etcd -y
```

```
vim /etc/etcd/etcd.conf
```

```
#[Member]
```

```
ETCD_DATA_DIR="/var/lib/etcd/default.etcd"
```

```
ETCD_LISTEN_PEER_URLS="http://192.168.26.91:2380,http://localhost:2380"
```

```
ETCD_LISTEN_CLIENT_URLS="http://192.168.26.91:2379,http://localhost:2379"
```

```
ETCD_NAME="default"
```

```
#[Clustering]
```

```
ETCD_ADVERTISE_CLIENT_URLS="http://localhost:2379"
```

```
systemctl start etcd
```

```
ls /var/lib/etcd/
```

```
etcdctl cluster-health
```

```
etcdctl member list
```

```
etcdctl ls /
```

```
etcdctl mkdir cka
```

```
etcdctl rmdir /cka
```

etcd集群



```
grep -o '^[^#].*' etcd.conf
ETCD_DATA_DIR="/var/lib/etcd/cluster.etcd"
ETCD_LISTEN_PEER_URLS="http://192.168.26.61:2380,http://localhost:2380"
ETCD_LISTEN_CLIENT_URLS="http://192.168.26.61:2379,http://localhost:2379"
ETCD_NAME="etcd-61"
ETCD_INITIAL_ADVERTISE_PEER_URLS="http://192.168.26.61:2380"
ETCD_ADVERTISE_CLIENT_URLS="http://localhost:2379,http://192.168.26.61:2379"
ETCD_INITIAL_CLUSTER="etcd-61=http://192.168.26.61:2380,etcd-62=http://192.168.26.62:2380,etcd-63=http://192.168.26.63:2380"
ETCD_INITIAL_CLUSTER_TOKEN="etcd-cluster"
ETCD_INITIAL_CLUSTER_STATE="new"
```



参数的意义

ETCD_NAME 节点名称，默认为default

ETCD_DATA_DIR 服务运行数据保存的路径

ETCD_LISTEN_PEER_URLS 监听的小伙伴通信的地址，比如http://ip:2380，如果有多个，使用逗号分隔。需要所有节点都能够访问，所以不要使用 localhost！

ETCD_LISTEN_CLIENT_URLS 监听的客户端服务地址

ETCD_ADVERTISE_CLIENT_URLS 对外公告的该节点客户端监听地址，这个值会告诉集群中其他节点。

ETCD_INITIAL_ADVERTISE_PEER_URLS 对外公告的该节点同伴监听地址，这个值会告诉集群中其他节点

ETCD_INITIAL_CLUSTER 集群中所有节点的信息，格式为

ETCD_INITIAL_CLUSTER_STATE 新建集群的时候，这个值为 new；假如加入已经存在的集群，这个值为 existing。

ETCD_INITIAL_CLUSTER_TOKEN 集群的ID，多个集群的时候，每个集群的ID必须保持唯一

etcdctl member add etcd-64 <http://192.168.26.64:2380>

etcd管理



```
export ETCDCTL_API=3
```

```
etcdctl put k1 vv1
```

```
etcdctl get k1
```

```
etcdctl snapshot save snap1.db
```

```
etcdctl del k1
```

```
etcdctl help snapshot save
```

```
etcdctl --cacert=domain1.crt --cert=node1.pem --key=node1.key -  
-endpoints=127.0.0.1:2379 snapshot save snap1.db
```

```
etcdctl --endpoints=127.0.0.1:2379 snapshot save snap1.db
```