CentOS7(mini) 安装 Kubernetes 集群（kubeadm方式）

安装CentOS

1. 安装**net-tools**

```
[root@localhost ~]# yum install -y net-tools
```

1. 关闭firewalld

```
[root@localhost ~]# systemctl stop firewalld && systemctl disable firewalld
Removed symlink /etc/systemd/system/multi-user.target.wants/firewalld.service.
Removed symlink /etc/systemd/system/dbus-org.fedoraproject.FirewallD1.service.
[root@localhost ~]# setenforce 0
[root@localhost ~]# sed -i 's/SELINUX=enforcing/SELINUX=disabled/g' /etc/selinux/config
```

安装Docker

> 如今Docker分为了Docker-CE和Docker-EE两个版本，CE为社区版即免费版，EE为企业版即商业版。我们选择使用CE版。

1. 安装yum源工具包

```
[root@localhost ~]# yum install -y yum-utils device-mapper-persistent-data lvm2
```

1. 下载docker-ce官方的yum源配置文件

```
[root@localhost ~]# yum-config-manager --add-repo https://download.docker.com/linux/centos/docker-ce.repo
```

1. 禁用docker-c-edge源配edge是不开发版，不稳定，下载stable版

```
yum-config-manager --disable docker-ce-edge
```

1. 更新本地YUM源缓存

```
yum makecache fast
```

1. 安装Docker-ce相应版本的

```
yum -y install docker-ce
```

1. 运行hello world

```
[root@localhost ~]# systemctl start docker
[root@localhost ~]# docker run hello-world
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
9a0669468bf7: Pull complete
Digest:
sha256:0e06ef5e1945a718b02a8c319e15bae44f47039005530bc617a5d071190ed3fc
Status: Downloaded newer image for hello-world:latest

Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
 1. The Docker client contacted the Docker daemon.
 2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
 3. The Docker daemon created a new container from that image which runs
the
    executable that produces the output you are currently reading.
 4. The Docker daemon streamed that output to the Docker client, which
sent it
    to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
 $ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
 https://cloud.docker.com/

For more examples and ideas, visit:
 https://docs.docker.com/engine/userguide/
```

安装kubelet与kubeadm包

> 使用kubeadm init命令初始化集群之下载Docker镜像到所有主机的实始化时会下载kubeadm必要的依赖镜像，同时安装etcd,kube-dns,kube-proxy,由于我们GFW防火墙问题我们不能直接访问，因此先通过其它方法下载下面列表中的镜像，然后导入到系统中，再使用kubeadm init来初始化集群

1. 使用DaoCloud加速器(可以跳过这一步)

```
[root@localhost ~]# curl -sSL
https://get.daocloud.io/daotools/set_mirror.sh | sh -s
http://0d236e3f.m.daocloud.io
docker version >= 1.12
{"registry-mirrors": ["http://0d236e3f.m.daocloud.io"]}
Success.
You need to restart docker to take effect: sudo systemctl restart docker
[root@localhost ~]# systemctl restart docker
```

1. 下载镜像,自己通过Dockerfile到dockerhub生成对镜像,也可以克隆我的

```
images=(kube-controller-manager-amd64 etcd-amd64 k8s-dns-sidecar-amd64
kube-proxy-amd64 kube-apiserver-amd64 kube-scheduler-amd64 pause-amd64 k8s-
dns-dnsmasq-nanny-amd64 k8s-dns-kube-dns-amd64)
for imageName in ${images[@]} ; do
  docker pull champly/$imageName
  docker tag champly/$imageName gcr.io/google_containers/$imageName
  docker rmi champly/$imageName
done
```

1. 修改版本

```
docker tag gcr.io/google_containers/etcd-amd64
gcr.io/google_containers/etcd-amd64:3.0.17 && \
docker rmi gcr.io/google_containers/etcd-amd64 && \
docker tag gcr.io/google_containers/k8s-dns-dnsmasq-nanny-amd64
gcr.io/google_containers/k8s-dns-dnsmasq-nanny-amd64:1.14.5 && \
docker rmi gcr.io/google_containers/k8s-dns-dnsmasq-nanny-amd64 && \
docker tag gcr.io/google_containers/k8s-dns-kube-dns-amd64
gcr.io/google_containers/k8s-dns-kube-dns-amd64:1.14.5 && \
docker rmi gcr.io/google_containers/k8s-dns-kube-dns-amd64 && \
docker tag gcr.io/google_containers/k8s-dns-sidecar-amd64
gcr.io/google_containers/k8s-dns-sidecar-amd64:1.14.2 && \
docker rmi gcr.io/google_containers/k8s-dns-sidecar-amd64 && \
docker tag gcr.io/google_containers/kube-apiserver-amd64
gcr.io/google_containers/kube-apiserver-amd64:v1.7.5 && \
docker rmi gcr.io/google_containers/kube-apiserver-amd64 && \
docker tag gcr.io/google_containers/kube-controller-manager-amd64
gcr.io/google_containers/kube-controller-manager-amd64:v1.7.5 && \
docker rmi gcr.io/google_containers/kube-controller-manager-amd64 && \
docker tag gcr.io/google_containers/kube-proxy-amd64
gcr.io/google_containers/kube-proxy-amd64:v1.6.0 && \
docker rmi gcr.io/google_containers/kube-proxy-amd64 && \
docker tag gcr.io/google_containers/kube-scheduler-amd64
gcr.io/google_containers/kube-scheduler-amd64:v1.7.5 && \
docker rmi gcr.io/google_containers/kube-scheduler-amd64 && \
docker tag gcr.io/google_containers/pause-amd64
gcr.io/google_containers/pause-amd64:3.0 && \
```

```
docker rmi gcr.io/google_containers/pause-amd64
```

1. 添加阿里源

```
[root@localhost ~]#  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
EOF
```

1. 查看kubectl kubelet kubeadm kubernetes-cni列表

```
[root@localhost ~]# yum list kubectl kubelet kubeadm kubernetes-cni
已加载插件：fastestmirror
Loading mirror speeds from cached hostfile
 * base: mirrors.tuna.tsinghua.edu.cn
 * extras: mirrors.sohu.com
 * updates: mirrors.sohu.com
可安装的软件包
kubeadm.x86_64                                                  1.7.5-0                                        kubernetes
kubectl.x86_64                                                  1.7.5-0                                        kubernetes
kubelet.x86_64                                                  1.7.5-0                                        kubernetes
kubernetes-cni.x86_64                                           0.5.1-0                                        kubernetes
[root@localhost ~]#
```

1. 安装kubectl kubelet kubeadm kubernetes-cni

```
[root@localhost ~]# yum install -y kubectl kubelet kubeadm kubernetes-cni
```

修改cgroups

```
vi /etc/systemd/system/kubelet.service.d/10-kubeadm.conf
```

> update KUBELET_CGROUP_ARGS=--cgroup-driver=systemd to KUBELET_CGROUP_ARGS=--cgroup-driver=cgroupfs

修改kubelet中的cAdvisor监控的端口，默认为0改为4194，这样就可以通过浏览器查看kubelet的监控cAdvisor的web页

```
[root@kub-master ~]# vi /etc/systemd/system/kubelet.service.d/10-kubeadm.conf
```

> Environment="KUBELET_CADVISOR_ARGS=--cadvisor-port=4194"

启动所有主机上的kubelet服务

```
[root@master ~]# systemctl enable kubelet && systemctl start kubelet
```

初始化master master节点上操作

```
[root@master ~]# kubeadm reset && kubeadm init --apiserver-advertise-address=192.168.0.100 --kubernetes-version=v1.7.5 --pod-network-cidr=10.200.0.0/16
[preflight] Running pre-flight checks
[reset] Stopping the kubelet service
[reset] Unmounting mounted directories in "/var/lib/kubelet"
[reset] Removing kubernetes-managed containers
[reset] Deleting contents of stateful directories: [/var/lib/kubelet /etc/cni/net.d /var/lib/dockershim /var/lib/etcd]
[reset] Deleting contents of config directories: [/etc/kubernetes/manifests /etc/kubernetes/pki]
[reset] Deleting files: [/etc/kubernetes/admin.conf /etc/kubernetes/kubelet.conf /etc/kubernetes/controller-manager.conf /etc/kubernetes/scheduler.conf]
[kubeadm] WARNING: kubeadm is in beta, please do not use it for production clusters.
[init] Using Kubernetes version: v1.7.5
[init] Using Authorization modes: [Node RBAC]
[preflight] Running pre-flight checks
[preflight] WARNING: docker version is greater than the most recently validated version. Docker version: 17.09.0-ce. Max validated version: 1.12
[preflight] Starting the kubelet service
[kubeadm] WARNING: starting in 1.8, tokens expire after 24 hours by default (if you require a non-expiring token use --token-ttl 0)
[certificates] Generated CA certificate and key.
[certificates] Generated API server certificate and key.
[certificates] API Server serving cert is signed for DNS names [master kubernetes kubernetes.default kubernetes.default.svc kubernetes.default.svc.cluster.local] and IPs [10.96.0.1 192.168.0.100]
[certificates] Generated API server kubelet client certificate and key.
[certificates] Generated service account token signing key and public key.
[certificates] Generated front-proxy CA certificate and key.
[certificates] Generated front-proxy client certificate and key.
[certificates] Valid certificates and keys now exist in "/etc/kubernetes/pki"
[kubeconfig] Wrote KubeConfig file to disk: "/etc/kubernetes/admin.conf"
[kubeconfig] Wrote KubeConfig file to disk: "/etc/kubernetes/kubelet.conf"
```

```
[kubeconfig] Wrote KubeConfig file to disk: "/etc/kubernetes/controller-manager.conf"
[kubeconfig] Wrote KubeConfig file to disk: "/etc/kubernetes/scheduler.conf"
[apiclient] Created API client, waiting for the control plane to become ready
[apiclient] All control plane components are healthy after 34.002949 seconds
[token] Using token: 0696ed.7cd261f787453bd9
[apiconfig] Created RBAC rules
[addons] Applied essential addon: kube-proxy
[addons] Applied essential addon: kube-dns

Your Kubernetes master has initialized successfully!

To start using your cluster, you need to run (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

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

You can now join any number of machines by running the following on each node
as root:

  kubeadm join --token 0696ed.7cd261f787453bd9 192.168.0.100:6443

[root@master ~]#
```

> **kubeadm join --token 0696ed.7cd261f787453bd9 192.168.0.100:6443** 这个一定要记住,以后无法重现，添加节点需要

添加节点

```
[root@node1 ~]# kubeadm join --token 0696ed.7cd261f787453bd9 192.168.0.100:6443
[kubeadm] WARNING: kubeadm is in beta, please do not use it for production clusters.
[preflight] Running pre-flight checks
[preflight] WARNING: docker version is greater than the most recently validated version. Docker version: 17.09.0-ce. Max validated version: 1.12
[preflight] WARNING: kubelet service is not enabled, please run 'systemctl enable kubelet.service'
[preflight] Starting the kubelet service
```

```
[discovery] Trying to connect to API Server "192.168.0.100:6443"
[discovery] Created cluster-info discovery client, requesting info from "https://192.168.0.100:6443"
[discovery] Cluster info signature and contents are valid, will use API Server "https://192.168.0.100:6443"
[discovery] Successfully established connection with API Server "192.168.0.100:6443"
[bootstrap] Detected server version: v1.7.10
[bootstrap] The server supports the Certificates API (certificates.k8s.io/v1beta1)
[csr] Created API client to obtain unique certificate for this node, generating keys and certificate signing request
[csr] Received signed certificate from the API server, generating KubeConfig...
[kubeconfig] Wrote KubeConfig file to disk: "/etc/kubernetes/kubelet.conf"

Node join complete:
* Certificate signing request sent to master and response
  received.
* Kubelet informed of new secure connection details.

Run 'kubectl get nodes' on the master to see this machine join.
```

在master配置kubectl的kubeconfig文件

```
[root@master ~]# mkdir -p $HOME/.kube
[root@master ~]# cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
[root@master ~]# chown $(id -u):$(id -g) $HOME/.kube/config
```

在Master上安装flannel

```
docker pull quay.io/coreos/flannel:v0.8.0-amd64
kubectl apply -f https://raw.githubusercontent.com/coreos/flannel/v0.8.0/Documentation/kube-flannel.yml
kubectl apply -f https://raw.githubusercontent.com/coreos/flannel/v0.8.0/Documentation/kube-flannel-rbac.yml
```

查看集群

```
[root@master ~]# kubectl get cs
NAME                 STATUS    MESSAGE              ERROR
scheduler            Healthy   ok
controller-manager   Healthy   ok
etcd-0               Healthy   {"health": "true"}
[root@master ~]# kubectl get nodes
NAME      STATUS     AGE       VERSION
```

```
master    Ready      24m       v1.7.5
node1     NotReady   45s       v1.7.5
node2     NotReady   7s        v1.7.5
[root@master ~]# kubectl get pods --all-namespaces
NAMESPACE     NAME                              READY     STATUS
   RESTARTS   AGE
kube-system   etcd-master                       1/1       Running
  0          24m
kube-system   kube-apiserver-master             1/1       Running
  0          24m
kube-system   kube-controller-manager-master    1/1       Running
  0          24m
kube-system   kube-dns-2425271678-h48rw         0/3       ImagePullBackOff
   0          25m
kube-system   kube-flannel-ds-28n3w             1/2       CrashLoopBackOff
   13         24m
kube-system   kube-flannel-ds-ndspr             0/2       ContainerCreating
  0          41s
kube-system   kube-flannel-ds-zvx9j             0/2       ContainerCreating
  0          1m
kube-system   kube-proxy-qxxzr                  0/1       ImagePullBackOff
   0          41s
kube-system   kube-proxy-shkmx                  0/1       ImagePullBackOff
   0          25m
kube-system   kube-proxy-vtk52                  0/1       ContainerCreating
  0          1m
kube-system   kube-scheduler-master             1/1       Running
  0          24m
[root@master ~]#
```

如果出现：The connection to the server localhost:8080 was refused - did you specify the right host or port?

> 解决办法： 为了使用kubectl访问apiserver，在~/.bash_profile中追加下面的环境变量： export KUBECONFIG=/etc/kubernetes/admin.conf source ~/.bash_profile 重新初始化kubectl