Kubernetes,

17Oct2022

Day10

K8s MultiNode Cluster Installation

Prepare Hosts to install K8s MultiNode Cluster

1MasterNode

-A compatible Linux host
-4GB or more of RAM per machine.
-2CPUs or more.
-Full network connectivity between all machines in the cluster (public or private network is fine).
-Unique hostname, MAC address, and product_uuid for every node.

2WorkerNode

-A compatible Linux host
-2GB or more of RAM per machine.
-1CPUs or more.
-Full network connectivity between all machines in the cluster (public or private network is fine).
-Unique hostname, MAC address, and product_uuid for every node.

after finish Installation power off host and make clone.
master1.example.com
worker1.example.com
worker2.example.com

MasterNode

hostnamectl set-hostname k8smaster1.example.com

ip a s
nmcli connection show
NAME UUID TYPE DEVICE
ens192 5bb5771d-b4df-4997-8d59-0e5c41fdb5b1 ethernet ens192
nmcli connection delete "ens192"
nmcli connection reload
nmcli connection add con-name "ens192" type ethernet autoconnect yes ifname ens192 ipv4.address "192.168.0.191/24" ipv4.dns "192.168.0.1" ipv4.gateway "192.168.0.1" +ipv4.dns "8.8.8.8" +ipv4.dns "8.8.4.4" ipv4.method manual
nmcli connection reload
vi /etc/hosts
192.168.0.191 k8smaster1.example.com k8smaster1
192.168.0.192 k8sworker1.example.com k8sworker1
192.168.0.193 k8sworker2.example.com k8sworker2
192.168.0.194 k8sworker3.example.com k8sworker3
:wq!

lsblk
sr0 11:0 1 10.7G 0 rom /media/cdrom
mkdir /media/cdrom
echo "/dev/sr0 /media/cdrom iso9660 defaults 0 0" >>/etc/fstab
mount -a
vi /etc/yum.repos.d/redhat.repo
[App]
name=AppStream
baseurl=file:///media/cdrom/AppStream/
gpgcheck=0
enabled=1
[BaseOS]
name=BaseOS
baseurl=file:///media/cdrom/BaseOS/
gpgcheck=0
enabled=1
:wq!
yum repolist
poweroff, take snapshot

WorkerNodes

hostname	k8sworker1.example.com k8sworker2.example.com k8sworker3.example.com
Config Network	192.168.0.192/24 192.168.0.193/24 192.168.0.194/24
Config Local DNS	192.168.0.191 k8smaster1.example.com k8smaster1 192.168.0.192 k8sworker1.example.com k8sworker1 192.168.0.193 k8sworker2.example.com k8sworker2 192.168.0.194 k8sworker3.example.com k8sworker3
Config Local Repository	/etc/yum.repos.d/rhel.repo

poweroff, make snapshot

Install K8s MultiNode Cluster

https://kubernetes.io/
https://kubernetes.io/docs/home/
https://kubernetes.io/docs/setup/production-environment/tools/kubeadm/install-kubeadm/

Master/Workers

Installing a container runtime

To run containers in Pods, Kubernetes uses a container runtime.

Container Runtimes

Option 1: From the official binaries

need to install a container runtime into each node in the cluster so that Pods can run there.

Kubernetes 1.25 requires that you use a runtime that conforms with the **Container Runtime Interface-CRI**

1- containerd

https://github.com/containerd/containerd/blob/main/docs/getting-started.md
https://github.com/containerd/containerd/releases

1-1. Installing containerd

```
# yum install wget -y
#    wget    https://github.com/containerd/containerd/releases/download/v1.6.8/containerd-1.6.8-linux-amd64.tar.gz
# ls containerd-1.6.8-linux-amd64.tar.gz
# tar Cxzvf /usr/local containerd-1.6.8-linux-amd64.tar.gz 1-
```

2. systemd

```
#  mkdir  -p  /usr/local/lib/systemd/system/
#  vi  /usr/local/lib/systemd/system/containerd.service
# Copyright The containerd Authors.
#
# Licensed under the Apache License, Version 2.0 (the "License");
# you may not use this file except in compliance with the License.
# You may obtain a copy of the License at
#
#     http://www.apache.org/licenses/LICENSE-2.0
#
# Unless required by applicable law or agreed to in writing, software
# distributed under the License is distributed on an "AS IS" BASIS,
# WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
# See the License for the specific language governing permissions and
# limitations under the License.

[Unit]
Description=containerd container runtime
Documentation=https://containerd.io
After=network.target local-fs.target

[Service]
#uncomment to enable the experimental sbservice (sandboxed) version of containerd/cri integration
#Environment="ENABLE_CRI_SANDBOXES=sandboxed"
ExecStartPre=-/sbin/modprobe overlay
ExecStart=/usr/local/bin/containerd

Type=notify
Delegate=yes
KillMode=process
Restart=always
RestartSec=5
# Having non-zero Limit*s causes performance problems due to accounting overhead
# in the kernel. We recommend using cgroups to do container-local accounting.
LimitNPROC=infinity
LimitCORE=infinity
LimitNOFILE=infinity
# Comment TasksMax if your systemd version does not supports it.
# Only systemd 226 and above support this version.
TasksMax=infinity
OOMScoreAdjust=-999

[Install]
WantedBy=multi-user.target
:wq!
# systemctl daemon-reload
# systemctl enable --now containerd
```

2- Installing runc

https://github.com/opencontainers/runc/releases

```
#    wget    https://github.com/opencontainers/runc/releases/download/v1.1.4/runc.amd64
# ls
containerd-1.6.8-linux-amd64.tar.gz  runc.amd64
# install -m 755 runc.amd64 /usr/local/sbin/runc
```

3- Installing CNI plugins

https://github.com/containernetworking/plugins/releases

```
#    wget    https://github.com/containernetworking/plugins/releases/download/v1.1.1/cni-plugins-linux-amd64-v1.1.1.tgz
# ls
cni-plugins-linux-amd64-v1.1.1.tgz  containerd-1.6.8-linux-amd64.tar.gz  runc.amd64
# mkdir -p /opt/cni/bin
# tar Cxzvf /opt/cni/bin cni-plugins-linux-amd64-v1.1.1.tgz

# sudo systemctl restart containerd
# systemctl status containerd
```

Kubernetes,

CRI-O

https://github.com/cri-o/cri-o/blob/main/install.md#readme

Operating system	$OS
Centos 8	CentOS_8
CRI-O	$VERSION
CRI-Ov1.23	1.23

curl -L -o /etc/yum.repos.d/devel:kubic:libcontainers:stable.repo https://download.opensuse.org/repositories/devel:/kubic:/libcontainers:/stable/CentOS_8/devel:kubic:libcontainers:stable.repo
curl -L -o /etc/yum.repos.d/devel:kubic:libcontainers:stable:cri-o:1.23.repo https://download.opensuse.org/repositories/devel:kubic:libcontainers:stable:cri-o:1.23/CentOS_8/devel:kubic:libcontainers:stable:cri-o:1.23.repo
yum install cri-o -y
yum install containernetworking-plugins ->it will install in CRI-O installation moment
systemctl daemon-reload
systemctl enable crio
systemctl start crio
systemctl status crio

Docker Engine

1- On each of your nodes, install **Docker** for your Linux distribution

to install Docker on RHEL **don't follow** https://docs.docker.com/engine/install/#server

to install Docker on RHEL **Follow** https://faun.pub/how-to-install-simultaneously-docker-and-podman-on-rhel-8-centos-8-cb67412f321e

dnf config-manager --add-repo=https://download.docker.com/linux/centos/docker-ce.repo
dnf config-manager --set-disabled docker-ce-stable
rpm --install --nodeps --replacefiles --excludepath=/usr/bin/runc https://download.docker.com/linux/centos/8/x86_64/stable/Packages/containerd.io-1.4.9-3.1.el8.x86_64.rpm
dnf install --enablerepo=docker-ce-stable docker-ce -y
sudo systemctl enable --now docker

or

systemctl enable docker.service
systemctl start docker.service
systemctl status docker.service

2- Install **cri-dockerd**, following the instructions in that source code repository

https://www.mirantis.com/blog/how-to-install-cri-dockerd-and-migrate-nodes-from-dockershim/

https://github.com/Mirantis/cri-dockerd/releases

wget https://github.com/Mirantis/cri-dockerd/releases/download/v0.2.6/cri-dockerd-0.2.6.amd64.tgz
tar xvf cri-dockerd-0.2.6.amd64.tgz
sudo mv ./cri-dockerd /usr/local/bin/
wget https://raw.githubusercontent.com/Mirantis/cri-dockerd/master/packaging/systemd/cri-docker.service
wget https://raw.githubusercontent.com/Mirantis/cri-dockerd/master/packaging/systemd/cri-docker.socket
mv cri-docker.socket cri-docker.service /etc/systemd/system/
sed -i -e 's,/usr/bin/cri-dockerd,/usr/local/bin/cri-dockerd,' /etc/systemd/system/cri-docker.service
systemctl daemon-reload
systemctl enable cri-docker.service
systemctl enable --now cri-docker.socket
systemctl status cri-docker.socket

Installing kubeadm, kubelet and kubectl

kubeadm:	the command to bootstrap the cluster.
kubelet:	the component that runs on all of the machines in your cluster and does things like starting pods and containers.
kubectl:	the command line util to talk to your cluster.

cat <<EOF | sudo tee /etc/yum.repos.d/kubernetes.repo
[kubernetes]
name=Kubernetes
baseurl=https://packages.cloud.google.com/yum/repos/kubernetes-el7-\$basearch
enabled=1
gpgcheck=1
gpgkey=https://packages.cloud.google.com/yum/doc/rpm-package-key.gpg
exclude=kubelet kubeadm kubectl
EOF
sudo setenforce 0
sudo sed -i 's/^SELINUX=enforcing$/SELINUX=permissive/' /etc/selinux/config

sudo yum install -y kubelet kubeadm kubectl --disableexcludes=kubernetes

sudo systemctl enable --now kubelet

Creating a cluster with kubeadm

Kubernetes sensitive on SWAP space, it should be **0**

free -m
swapoff -a
free -m

on MasterNode

hostname -i
192.168.0.191

Note: If kubelet is not running them user below four commands

1. sudo –i
2. swapoff –a
3. exit
4. strace -eopenat kubectl version

kubeadm init --apiserver-advertise-address=192.168.0.191 --cri-socket=/var/run/crio/crio.sock --v=5
.
.
.
Your Kubernetes control-plane has initialized **successfully**!
mkdir -p $HOME/.kube
sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
sudo chown $(id -u):$(id -g) $HOME/.kube/config

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

kubeadm join 192.168.0.191:6443 --token 7qu88a.4eqvd23ahd8yauus --discovery-token-ca-cert-hash sha256:6cc0c1339fbb546f658d23d091669483acc461533cca1befe17f9673c15e3163

on WorkerNodes

kubeadm join 192.168.0.191:6443 --token 7qu88a.4eqvd23ahd8yauus --discovery-token-ca-cert-hash sha256:6cc0c1339fbb546f658d23d091669483acc461533cca1befe17f9673c15e3163 --cri-socket=/var/run/crio/crio.sock --v=5

NOTE: token doesn't work more than 23Hrs

on MasterNode

kubectl get nodes

NAME	STATUS	ROLES	AGE	VERSION
k8smaster1.example.com	Ready	control-plane	5m32s	v1.25.3
k8sworker1.example.com	Ready	<none>	102s	v1.25.3
k8sworker3.example.com	Ready	<none>	112s	v1.25.3

kubeadm completion bash >>~/.bashrc
kubectl completion bash >>~/.bashrc
echo "autocmd FileType yaml setlocal ai ts=2 sw=2 et cursorcolumn" >~/.vimrc