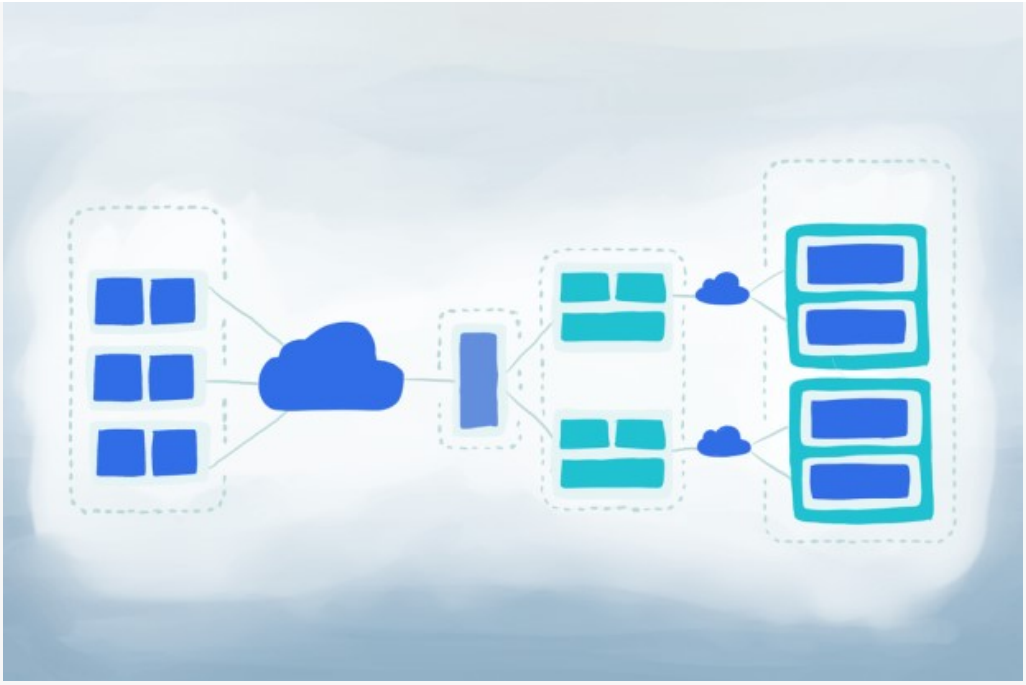
Kubernetes cluster setup using virtual machines



Preface

Recently I needed to setup k8s cluster on my local machine in order to see if I'm able to do it on bare-metal servers lately. So I decided to first give it a shot using Virtual Box machines. There were a few places where I was a bit stuck and spent some times googling. This article should summarize all of the problems and give you solution.

Most of knowledge used here can be found in official kubernetes documentation. I will also show you how to configure your virtual machines and their network in order to work well with k8s cluster.

If you are new to VirtualBox I suggest to visit official documentation as well as CentOS 7 specific tutorial:

- https://www.virtualbox.org/wiki/Documentation
- https://wiki.centos.org/HowTos/Virtualization/VirtualBox/CentOSguest

Installing and configuring Virtual Box VMs

Our goal is to setup following infrastructure:

Machine name	Hostname	IP
kubemaster	kubemaster	192.168.1.20
kubeslave1	kubeslave1	192.168.1.21
kubeslave2	kubeslave2	192.168.1.22

VirtualBox network configuration

First we need to install VirtualBox. On Ubuntu it's a simple `sudo apt install virtualbox`command.

Now we are going to configure our host-only network, which will be used inside of VirtualBox environment. Open VirtualBox, go to **File -> Preferences -> Network -> Host-only Network**. If there is nothing configured yet, add new Host Only network(if you already have something configured, you need to either change IP as in this tutorial or make note what IPs to use later). You can now disable DHCP inside of the settings and also set IPv4 address to 192.168.99.1.

Once you have this configured, let's create some VMs.

Create base virtual machine

Given all our machines will need the same base pieces of sw installed before setting up k8s, we are going to create just one VM now and clone the others later. We will use CentOS 7 as a OS for all VMs(K8s official tutorial uses them aswell, so you should be fine with this choice). Get ISO from https://www.centos.org/download/ . Minimal ISO is perfectly fine for our purpose. Create new virutal machine, called kubemaster, with at least following resources:

- 1 CPU

- 2 GB RAM

Assign host only network to second adapter and add downloaded CentOS iso file to storage.

Boot up the machine and install CentOS. In this tutorial, I'm using root accounts only on all 3 VMs. This is not recommended, but we will do it just to make it simpler now.

Once you have your system running, do following:

- Update packages

```
yum update
```

- Install wget

```
yum install wget
```

- Set IP address for host only network with

```
nmtui
```

  (usually it's sitting on enp8s0 adapter)

- Set hostname

```
hostnamectl set-hostname kubemaster
```

- Disable firewall

```
systemctl disable firewalld && systemctl stop firewalld
```

- Disable Selinux – http://idroot.net/tutorials/how-to-disable-selinux-on-centos-7/
- Add following records to /etc/hosts:
  - 192.168.99.20 kubemaster.test.com kubemaster
  - 192.168.99.21 kubeslave1.test.com kubeslave1
  - 192.168.99.22 kubeslave2.test.com kubeslave2

Now you are ready to install base kubernetes pieces. This is from k8s tutorial mentioned above:

```
cat < /etc/yum.repos.d/kubernetes.repo [kubernetes]
name=Kubernetes
baseurl=http://yum.kubernetes.io/repos/kubernetes-el7-x86_64
enabled=1 gpgcheck=1 repo_gpgcheck=1
gpgkey=https://packages.cloud.google.com/yum/doc/yum-key.gpg
https://packages.cloud.google.com/yum/doc/rpm-package-key.gpg
EOF setenforce 0 yum install -y docker kubelet kubeadm kubectl
kubernetes-cni systemctl enable docker && systemctl start
docker systemctl enable kubelet && systemctl start kubelet
```

This should install all dependencies we need for k8s cluster.

Clone machines

Now create a linked clone machines from kubemaster machines created before. Once you're done, boot into machine and change following things to match infrastructure:

- Set IP address 192.168.99.21 (or 22 for second slave) for host only network.
- Set hostname `hostnamectl set-hostname kubeslave1 (or kubeslave2 for second slave)` Everything else is already configured.

Setup Kubernetes cluster

Now we are going to setup whole cluster in just few steps.

Boot all 3 machines up.

Now we need to ensure that hostname matches our host only network ip. You can do it with `hostname -i` and it should return IP of machine. If that's not the case, just try to restart VMs or check your network configuration.

Now on kubemaster, run following set of commands:

- Get jq utility

```
wget https://github.com/stedolan/jq/releases/download/jq-1.5/jq-linux64 && mv jq-linux64 /usr/bin/jq && chmod +x /usr/bin/jq
```

- Initialize cluster and copy connect command

```
kubeadm init --api-advertise-address=192.168.99.20
```

- Workaround for kube-dns problem with multiple network interfaces(run on master node):

```
kubectl -n kube-system get ds -l "component=kube-proxy" -o json | jq ".items[0].spec.template.spec.containers[0].command |= .+[\"--proxy-mode=userspace\"]" | kubectl apply -f - && kubectl -n kube-system delete pods -l "component=kube-proxy"
```

- Install network layer for k8s

```
kubectl apply -f https://git.io/weave-kube
```

- Connect every node with command saved from kubeadm init

And that is it. To ensure that your cluster is OK, try following:

```
[root@kubemaster ~] # kubectl get nodes

NAME         STATUS         AGE

kubemaster   Ready,master   1h

kubeslave1   Ready          1h

kubeslave2   Ready          1h

[root@kubemaster ~]# kubectl get pods --all-namespaces

NAMESPACE     NAME                                    READY
STATUS    RESTARTS   AGE

kube-system   dummy-2088944543-mfrw9                  1/1
Running   0          1h

kube-system   etcd-kubemaster                         1/1
Running   0          1h

kube-system   kube-apiserver-kubemaster               1/1
Running   5          1h

kube-system   kube-controller-manager-kubemaster      1/1
Running   0          1h

kube-system   kube-discovery-1769846148-pjdlm         1/1
Running   0          1h

kube-system   kube-dns-2924299975-b9nqt               4/4
Running   0          1h
```

```
kube-system   kube-proxy-4lltt                        1/1
Running   0          1h


kube-system   kube-proxy-8qhm6                        1/1
Running   0          1h


kube-system   kube-proxy-dgxgj                        1/1
Running   0          1h


kube-system   kube-scheduler-kubemaster               1/1
Running   0          1h


kube-system   weave-net-8tnbk                         2/2
Running   1          1h


kube-system   weave-net-b6fdl                         2/2
Running   0          1h


kube-system   weave-net-f59vj                         2/2
```