## Getting started with Kubernetes

## Introduction:

The end goal is to build a convenient, scalable framework for deploying applications such as Zeppelin, Spark, and TensorFlow easily. Additionally, this system will support redundancy and allow us to focus more on application development, rather than system deployment.

## Kubernetes:

Kubernetes

## The Setup

### 1. Setting up The Host Name For Our Virtual Machine

In order for our nodes (machines) to communicate properly, we will need to set up unique host names for each virtual machine in the cluster.

1. Open up the terminal in your ubuntu VM
2. Run the command sudo hostname <insert creative name here>

It21-Virtual

1. Edit the hostname file by running sudo nano /etc/hostname

* Change the content here to whatever you put in step 1B
* Save by pressing ctrl+x, and y to confirm

1. Edit the host file to match the updated change by running sudo nano /etc/hosts

* Change the value that was originally present in step 1C to the new host name that you created in 1B

1. Verify that your host was set correctly by running hostname
2. Reboot your system by running sudo reboot now

### 2. Preparing Our Docker Image

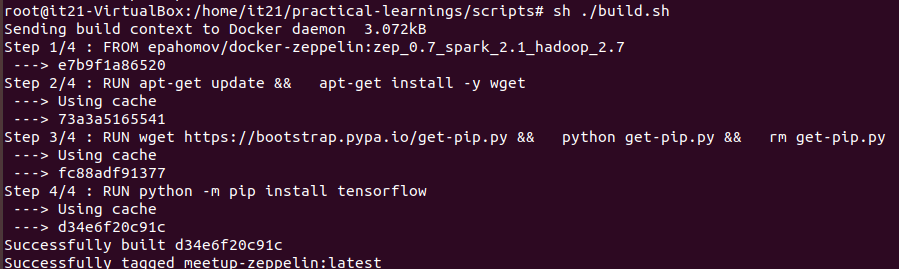
We’ll need our custom zeppelin docker image to get things running.

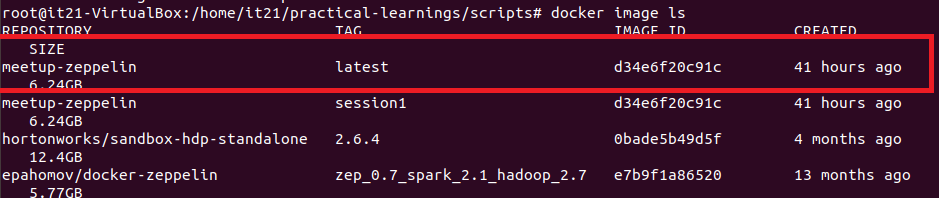
1. Open your terminal
2. Run cd ~/practical-learnings/scripts

* Or a different location if you cloned our project elsewhere

1. Run sh ./build.sh

docker build -t meetup-zeppelin .





### 3. Installing Required Software

In order to proceed, we will need to install some new software. A script has been provided for your convenience to make this process easier.

Run sudo sh ./install-kubernetes.sh

./install-kubernetes.sh：

apt-get install -y apt-transport-https ca-certificates curl software-properties-common

curl -fsSL https://download.docker.com/linux/ubuntu/gpg | apt-key add -

add-apt-repository \

"deb [arch=amd64] https://download.docker.com/linux/ubuntu \

$(lsb\_release -cs) \

stable"

apt-get update

apt-get install -y docker-ce=17.03.0~ce-0~ubuntu-xenial

apt-get update && apt-get install -y apt-transport-https

curl -s https://packages.cloud.google.com/apt/doc/apt-key.gpg | apt-key add -

cat <<EOF >/etc/apt/sources.list.d/kubernetes.list

deb http://apt.kubernetes.io/ kubernetes-xenial main

EOF

apt-get update

apt-get install -y kubelet=1.6.7-00 kubeadm=1.6.7-00 kubectl=1.6.7-00 kubernetes-cni

### 4. Preparing the data

In order for our Zeppelin application to access our notebooks and data sets, we will need to create a couple of folders and place files into them. These will be the notebooks and data from session 1

1. Run sudo mkdir -p /var/lib/zeppelin
2. Run sudo cp -R ./notebook /var/lib/zeppelin/
3. Run sudo cp -R ./data-sets /var/lib/zeppelin/

### 5. Master Setup (Do This If You Are The Master Node)

The master node will need to run the following setup in order to start up the cluster and deploy applications.

1. We will need to become root for this, so run sudo su
2. Run kubeadm init --token 90f324.dfa440add2b1bd93 --pod-network-cidr 10.244.0.0/16

* This will initialize our kubernetes cluster
* It may be useful to copy the final output from this command that tells worker nodes how to join the cluster

1. Run export KUBECONFIG=/etc/kubernetes/admin.conf
2. Run kubectl create -f canal.yaml
3. If you missed out in the join information from step 4E, you can now use ifconfig to fetch your ip address

* The format should be similar to `addr:192.168.XXX.XX` and will be located near an adapter that looks like enp7s0 or enp3s0
* If you can’t find an adapter with an address similar to the above, you may not have configured your virtual machine network adapter to use bridge mode. Please refer to the meetup preparation docs on how to do this

1. After the worker nodes have joined, you will be able to list them by calling kubectl get nodes

* They should eventually all show up in READY state after a few minutes. If not you may have to look to see if each machine is configured correctly for network as in step 4H

1. To see detailed status of what is going on in the system, you can run kubectl get pods -o wide --all-namespaces
2. Finally, to deploy our spark application, run kubectl create -f zeppelin.yaml

If this does not work:

You need run

kubeadm reset

rm -r /etc/kubenetes/

rm -r /var/lib/kubelet

systemctl stop kubelet

swapoff -a

free -mh

update the version in install-kubernetes.sh

apt-get install -y kubelet = 1.9.7-00 kubeadm=1.9.7-00 kub$

### 6. Worker Setup (Do This If You Are A Worker Node)

The master node will need to run the following setup in order to start up the cluster and deploy applications.

1. We will need to become root for this, so run sudo su
2. Run kubeadm join --token 90f324.dfa440add2b1bd93 <ipaddressofmaster>:6443
3. Run export KUBECONFIG=/etc/kubernetes/kubelet.conf

* You can now use kubectl, similar to the master, to get cluster information. However, you cannot modify anything about the cluster, such as deploying applications

### 7. Cleanup

This is an important step if you don’t want Kubernetes to use a lot of resources while you use you VM.

1. As root, run kubeadm reset