Underline = Names/words/numbers of settings in code or elsewhere

**Bold = actual lines of code**

*Italic = Manual actions*

*Execute on all nodes*

<https://askubuntu.com/questions/22835/how-to-network-two-ubuntu-computers-using-ethernet-without-a-router>

**Master**

IP 10.42.0.1

Netmask 255.255.255.0

MAC adress 00:1e:06:42:02:17

UUID=e139ce78-9841-40fe-8823-96a304a09859

**Slave1**

IP 10.42.0.208

Netmask 255.255.255.0

MAC adress 00:1e:06:42:92:22

UUID=e139ce78-9841-40fe-8823-96a304a09859

**Slave 2**

IP 10.0.0.2

Netmask 255.255.255.0

MAC adress 00:1e:06:42:92:22

Slave 3

DOE ONDERSTAANDE STAPPEN TOT DE STREEP VOOR ELKE INDIVIDUELE ODROID

Setting up the basics (source :<https://www.polarsparc.com/xhtml/N2-Cluster.html>)

Step 1: *Open* Control center*, click* Software updater

Step 2: *Perform reboot*

**$ sudo** **Reboot**

Step 3: *Open* Control center, *type* time, *change time to* ”local”

Step 4: *Open* Control center, *type* User, *change password to Brouwerx where x stands for the ID (master = 1, slave = 2 to y)*

Step 5: *Open* Mate Terminal, *Set* Permit root login to “no” (Specifies whether root can log in using ssh(1), makes it more secure?). *Save* after

**$ sudo** **pluma /etc/ssh/sshd\_config**

Step 6: Name each Odroid. *Save* after

(1) “MasterNode” for Master node

(2) “SlaveNodeX” for Slave node x = 1-3

**$ sudo** **pluma /etc/hostname**

Step 7: Set the IP, networks gateway manually. *Open* *control center*, *Open* *network connection*, *Double click* Wired connection 1, *select* IPv4 Settings, *select* Manual and Add

*Set* *IP address*

(1) master node as IP to 192.168.1.51

(2) slave nodes as IP to 192.168.1.(51+x)

*Set* *network* to 255.255.255.0

*Set* *gateway* to 192.168.1.1

*Set* *DNS servers* to 8.8.8.8,8.8.4.4

**-------------------------------------------------------**

**EXTRA STEP, REMOVE IDENTIFIERS (**[**https://magazine.odroid.com/article/kubernetes-on-an-odroid-n2-cluster/**](https://magazine.odroid.com/article/kubernetes-on-an-odroid-n2-cluster/)**) FROM EVERY NODE, THEY ARE THE SAME**

Step 8: *Download* docker from repositories (source:https://docs.docker.com/engine/install/ubuntu/) (source:<https://magazine.odroid.com/article/kubernetes-on-an-odroid-n2-cluster/>)

**$ sudo** **apt-get update**

**$ sudo** **apt-get upgrade**

**$ sudo** **apt-get install apt-transport-https**

**$ sudo** **apt-get install ca-certificates**

**$ sudo** **apt-get install curl**

**$ sudo** **apt-get install gnupg**

**$ sudo** **apt-get install lsb-release**

**OF**

**sudo apt-get remove docker docker-engine docker.io containerd runc**

**sudo apt-get update**

**sudo apt-get install \**

**apt-transport-https \**

**ca-certificates \**

**curl \**

**gnupg \**

**lsb-release**

Step 9: **Download vanaf website** [**https://download.docker.com/linux/ubuntu/dists/focal/pool/stable/arm64/**](https://download.docker.com/linux/ubuntu/dists/focal/pool/stable/arm64/) **“containerd.io” (**1.4.12-1**) “docker-ce-cli”(** 20.10.12-3**) en “docker-ce” (**20.10.12-3**) (nadeel – geen automatische updates, je download de laatste versies).**

**Plaats files in de download map en verander de namen om files te transferren naar andere nodes.**

scp -r /home/odroid/Downloads/containerd.io.deb odroid@10.0.0.1:/home/odroid/Downloads

scp -r /home/odroid/Downloads/docker-ce-cli.deb odroid@10.0.0.1:/home/odroid/Downloads

scp -r /home/odroid/Downloads/docker-ce.deb odroid@10.0.0.1:/home/odroid/Downloads

Step 10: **Run de packages vanuit downloads om deze te installeren**

**$ sudo** **dpkg -i /home/odroid/Downloads/containerd.io.deb**

**$ sudo** **dpkg -i /home/odroid/Downloads/docker-ce-cli.deb**

**$ sudo** **dpkg -i /home/odroid/Downloads/docker-ce.deb**

**HEB GROEN GEMARKEERDE SIMPELWEG IN TERMINAL GEDAAN EN DIT MAAKTE DE JSON FILE EN GAF DE GOEDE DRIVER CGROUP AAN MET**

**sudo docker info**

[**https://medium.com/javarevisited/docker-steps-for-kubernetes-k8s-2-a1ed00acdeba**](https://medium.com/javarevisited/docker-steps-for-kubernetes-k8s-2-a1ed00acdeba)

Step 14: Maak een JSON file genaamd /etc/docker/**daemon.json** voor gebruik van 1 systemd als hoofdnode ipv cgroupsfs (kubernetes werkt hierdoor beter)

**$ sudo** **Pluma /etc/docker/daemon.json**

**Copy-paste de volgende commands**

**sudo mkdir /etc/docker**

**cat <<EOF | sudo tee /etc/docker/daemon.json**

**{**

**"exec-opts": ["native.cgroupdriver=systemd"],**

**"log-driver": "json-file",**

**"log-opts": {**

**"max-size": "100m"**

**},**

**"storage-driver": "overlay2"**

**}**

**EOF**

**Restart docker for config to take place**

**$ sudo** **mkdir -p /etc/systemd/system/docker.service.d**

**$ sudo** **systemctl daemon-reload**

**$ sudo** **systemctl restart docker**

**Reboot**

sudo reboot now

(<https://www.howtoforge.com/tutorial/how-to-create-docker-images-with-dockerfile/> )

**CREATING A DOCKER IMAGE FROM A DOCKERFILE**

**Create a directory**

mkdir -p R-docker;

**Navigate to this directory**

cd R-docker/

**Create a Dockerfile in this directory and fill with information**

touch Dockerfile

pluma Dockerfile

**Build an image of this docker file located in**

docker build -t Vincent/Docker-test-v1 /home/odroid/R-docker

Download docker images which it is dependent on?

docker pull rocker/r-ver:4.0.3

**See all images**

docker image ls

**Run the image**

docker run Vincent/Docker-test-v1

[**https://mdneuzerling.com/post/hosting-a-plumber-api-with-kubernetes/**](https://mdneuzerling.com/post/hosting-a-plumber-api-with-kubernetes/)

[**https://www.r-bloggers.com/2021/05/best-practices-for-r-with-docker/**](https://www.r-bloggers.com/2021/05/best-practices-for-r-with-docker/)

(<https://www.howtoforge.com/tutorial/how-to-create-docker-images-with-dockerfile/> )

Step 17: Prepareer voor Kubernetes download (<https://kubernetes.io/docs/setup/production-environment/tools/kubeadm/install-kubeadm/#installing-docker> )

**- Configure sysctl voor zien van bridged traffic en enable br\_netfilter**

**$ sudo** **cat <<EOF | sudo tee /etc/modules-load.d/k8s.conf**

**br\_netfilter**

**EOF**

**+**

**$ sudo** **cat <<EOF | sudo tee /etc/sysctl.d/k8s.conf**

**net.bridge.bridge-nf-call-ip6tables = 1**

**net.bridge.bridge-nf-call-iptables = 1**

**EOF**

sudo sysctl --system

lsmod | grep br\_netfilter

**- Open sws port 10259, 10257,10250 (**[**https://linuxconfig.org/how-to-open-allow-incoming-firewall-port-on-ubuntu-20-04-focal-fossa**](https://linuxconfig.org/how-to-open-allow-incoming-firewall-port-on-ubuntu-20-04-focal-fossa) **) ook 6443, 2379, 2380, 10251, 10252, 30000, 32767**

sudo ufw allow from any to any port 10259,10257,10250,6443,2379,2380,10251,10252,30000,32767,3000,8787 proto tcp

$ sudo ufw allow from any to any port 10259 proto tcp

Step 16: Installeer Kubernetes via reposatory (Source:<https://magazine.odroid.com/article/kubernetes-on-an-odroid-n2-cluster/>)

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

sudo curl -fsSLo /usr/share/keyrings/kubernetes-archive-keyring.gpg https://packages.cloud.google.com/apt/doc/apt-key.gpg

echo "deb [signed-by=/usr/share/keyrings/kubernetes-archive-keyring.gpg] https://apt.kubernetes.io/ kubernetes-xenial main" | sudo tee /etc/apt/sources.list.d/kubernetes.list

sudo apt-get update

sudo apt-get install -y kubelet kubeadm kubectl

sudo apt-mark hold kubelet kubeadm kubectl docker-ce

**$ sudo** **apt-get update**

**Sudo Tee maakt een nieuwe file aan met de link beschreven onder deb. Blijkbaar is dit anders dan de normale sources.list apt file, maar wordt wel gebruikt voor updates.**

**$ sudo** **apt-get install -y kubeadm**

**$ sudo** **apt-get install -y kubelet kubectl**

**$ sudo** **apt-mark hold kubelet kubeadm kubectl docker-ce**

**makes sure these versions of the programs aren’t updated.**

**Swap disabled. You MUST disable swap in order for the kubelet to work properly. (**[**https://linuxconfig.org/how-to-install-kubernetes-on-ubuntu-20-04-focal-fossa-linux**](https://linuxconfig.org/how-to-install-kubernetes-on-ubuntu-20-04-focal-fossa-linux) **)**

**$ sudo** **swapoff -a**

**Maak een kubernetes cluster !**

sudo kubeadm init

sudo kubeadm init --pod-network-cidr=192.168.0.0/16 \*alleen in combo met calico

**Voorbereiding voor cluster gebruik en** **krijg toestemming om kubectl op cluster te runnen als non-root**

mkdir -p $HOME/.kube

sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config

sudo chown $(id -u):$(id -g) $HOME/.kube/config

**kijk of alle ready is (dns servers nog niet online, is na podnetwork deploy wel online)**

kubectl get nodes

kubectl get pods -n kube-system -o wide

**gooi pod network los**

kubectl apply -f "https://cloud.weave.works/k8s/net?k8s-version=$(kubectl version | base64 | tr -d '\n')"

**zie trafic in pods (wacht 33 sec**

kubectl get pods -n kube-system -l name=weave-net -o wide

kubectl logs -n kube-system weave-net-x25wh weave

**Join de cluster met de andere computers**

kubeadm join 192.168.1.229:6443 --token 4zfn2c.vdjlvsul4gvxuo4l \  
 --discovery-token-ca-cert-hash sha256:54d32594f36395ec7c5f146f00a0a181af005090f31d17bbe24f8207f553e705

**check of er geen overla is in machine id**

kubectl get pods -n kube-system -l name=weave-net -o wide

kubectl logs -n kube-system weave-net-7nr25 weave

**Check of de andere nodes aanwezig zijn op de cluster**

kubectl get nodes -o wide

kubectl get pods --all-namespaces -o wide

**Laat zeer specifieke informatie zien over een bepaalde node (nu de masternode)**

kubectl describe pod kube-controller-manager-masternode -n kube-system

kubectl describe pod kube-proxy-hnrl5 -n kube-system

**TROUBLE SHOOTING**

sudo **kubeadm reset**

**sudo apt-get purge kubeadm kubectl kubelet kubernetes-cni kube\***

**sudo apt-get autoremove**

**sudo** **rm -rf /etc/cni /etc/kubernetes /var/lib/dockershim /var/lib/etcd /var/lib/kubelet /var/run/kubernetes ~/.kube/\* ~/.kube**

sudo iptables -F && sudo iptables -t nat -F && sudo iptables -t mangle -F && sudo iptables -X

sudo systemctl daemon-reload

sudo systemctl restart docker

sudo apt-get install -y kubelet kubeadm kubectl

sudo apt-mark hold kubelet kubeadm kubectl docker-ce

sudo swapoff -a

**reboot**

sudo sysctl --system

lsmod | grep br\_netfilter

**trouble shooting**

kubectl drain slavenode1 --ignore-daemonsets --force

kubectl drain slavenode1 --force

kubectl cordon slavenode1

kubectl delete pod rstudio

**be sure to uncordon slavenode 1 later to remove Scheduling disabled**

kubectl uncordon slavenode1

**Voorwaardes om aan te voldoen**

**(source: https://kubernetes.io/docs/setup/production-environment/tools/kubeadm/install-kubeadm/#installing-docker)**

**A compatible Linux host. The Kubernetes project provides generic instructions for Linux distributions based on Debian and Red Hat, and those distributions without a package manager.**

**2 GB or more of RAM per machine (any less will leave little room for your apps).**

**2 CPUs 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. See here for more details.**

**Certain ports are open on your machines. See here for more details.**

**Swap disabled. You MUST disable swap in order for the kubelet to work properly. (**[**https://linuxconfig.org/how-to-install-kubernetes-on-ubuntu-20-04-focal-fossa-linux**](https://linuxconfig.org/how-to-install-kubernetes-on-ubuntu-20-04-focal-fossa-linux) **)**