3.2 Readme - Group28

1.Install the docker

**Use ssh to access to the VM:**

ssh [stude](mailto:student053@145.100.134.52)[nt037@145.100.134.37](mailto:nt037@145.100.134.37) (student037-fieb7eaRie3thohl)

ssh [stude](mailto:student053@145.100.134.52)[nt038@145.100.134.38](mailto:nt038@145.100.134.387) (student038-bae0Sheiphoopiem)

ssh [stude](mailto:student053@145.100.134.52)[nt039@145.100.134.39](mailto:nt039@145.100.134.39) (student039-eediSiengaishoo5)

**Update the package list:**

sudo apt update

**Install packages:**

sudo apt install apt-transport-https ca-certificates curl gnupg2 software-properties-common

**Add docker GPG key in the system:**

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

**Add docker repository to APT:**

sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/debian $(lsb\_release -cs) stable"

**# If has the error “add-apt-repository:command not found”**

sudo apt-get install python-software-properties

sudo apt-get install software-properties-common

Update the docker list again:

sudo apt update

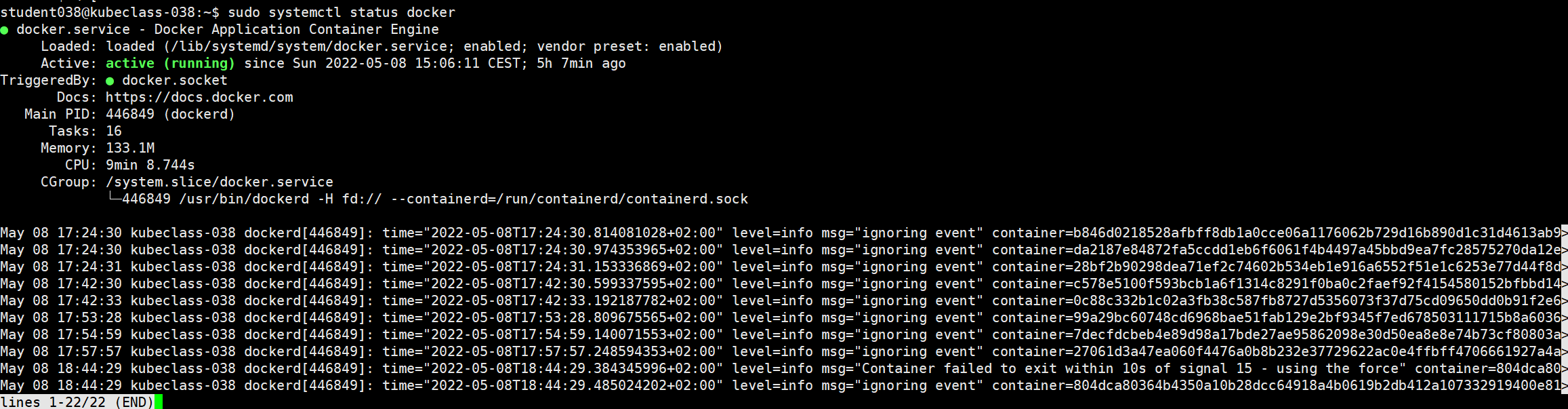
**Install docker:**

sudo apt-get -y install docker-ce docker-ce-cli containerd.io

sudo systemctl enable docker

**# Check the status of docker:**

sudo systemctl status docker



2.Install the k8s

**Install extra dependencies**

sudo apt-get -y install apt-transport-https net-tools

**Add the GPG key for Kubernetes repository**

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

**Add kubernetes repository to APT:**

sudo apt-add-repository "deb http://apt.kubernetes.io/ kubernetes-xenial main"

**Update the repository again:**

sudo apt-get update -y

**Install kubernetes with apt-get:**

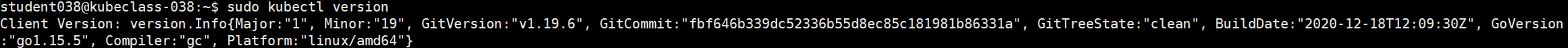
sudo apt-get install kubelet=1.19.6-00 kubeadm=1.19.6-00 kubectl=1.19.6-00

**Hold the packages:**

sudo apt-mark hold kubelet kubectl kubeadm

**# Check the version of Kubernetes:**

sudo kubectl version

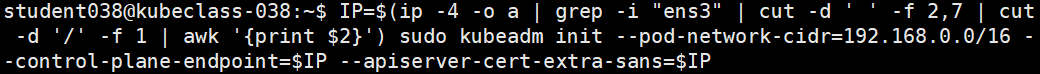


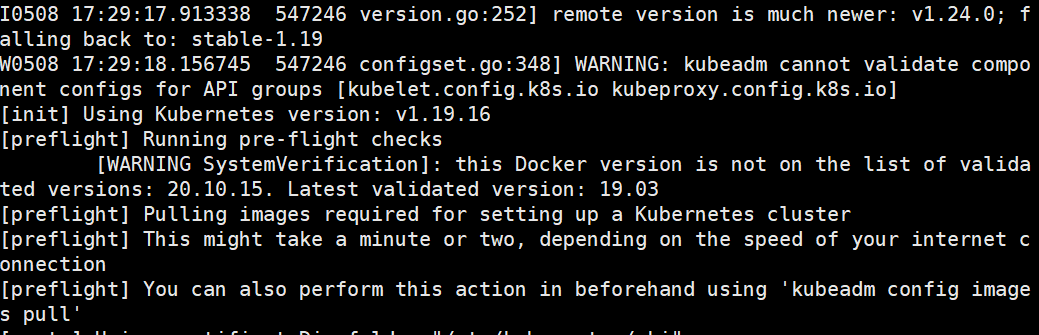
**Autostart:**

sudo systemctl enable kubelet

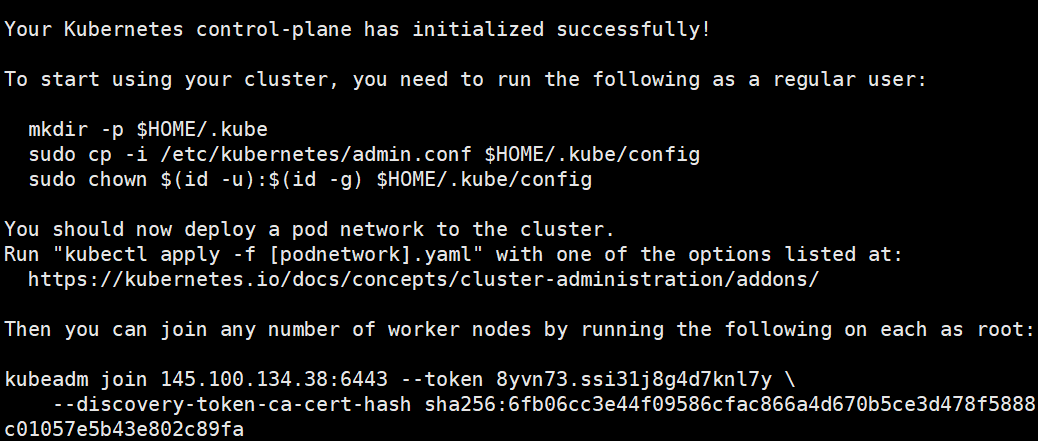
3.Control node 038

**Use kubeadm to setup the control plane：**





·····································



**We can get the token:**

kubeadm join 145.100.134.38:6443 --token 8yvn73.ssi31j8g4d7knl7y --discovery-token-ca-cert-hash sha256:6fb06cc3e44f09586cfac866a4d670b5ce3d478f5888c01057e5b43e802c89fa

**Setting up your cluster:**

To start using your cluster, you need to run the following 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

**For Calico it requires two YAML configuration files to be   
applied:**

kubectl create -f   
https://docs.projectcalico.org/manifests/tigera-operator.yaml

kubectl create -f   
https://docs.projectcalico.org/manifests/custom-resources.yaml

The Kubernetes cluster has been set up.

**Writes the new commands to the /var/lib/kubelet/kubeadm-flags.env**

**File:**

sudo echo KUBELET\_KUBEADM\_ARGS=\"--network-plugin=cni --pod-infra-container-image=k8s.gcr.io/pause:3.2 --node-ip=$(ip -4 -o a | grep -i "ens3" | cut -d ' ' -f 2,7 | cut -d '/' -f 1 | awk '{print $2}')\" | sudo tee /var/lib/kubelet/kubeadm-flags.env

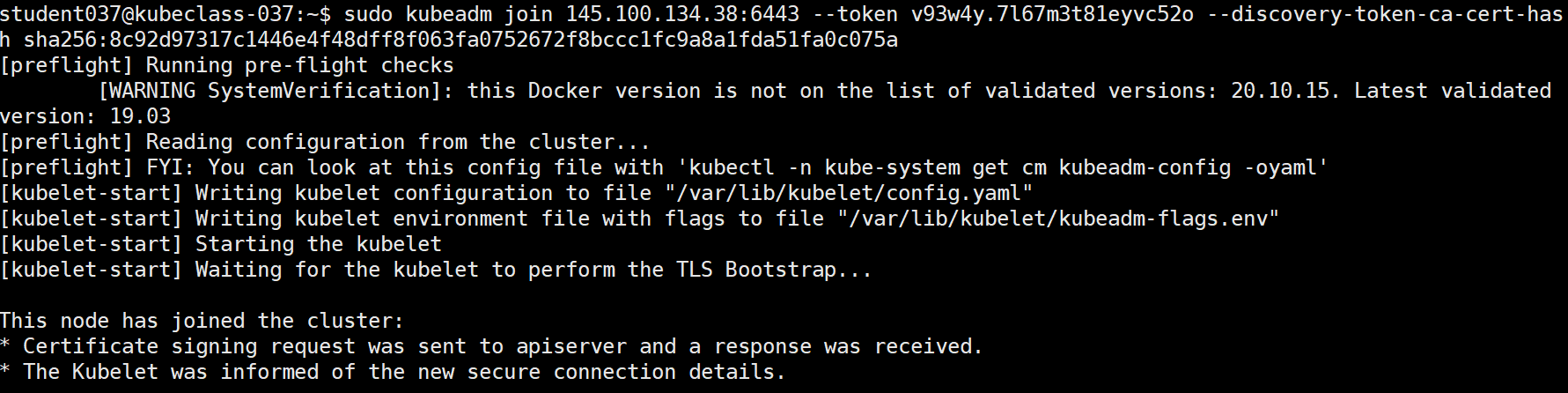
**Restarts the kubelet service so it uses those argument:**

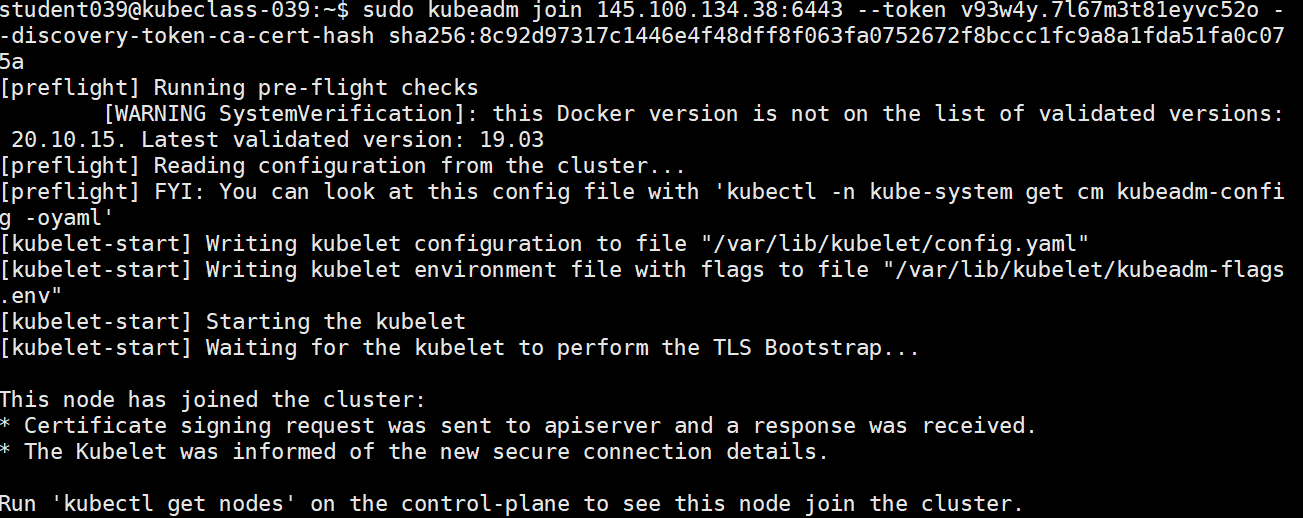
sudo systemctl restart kubelet.service

4 Worker nodes 037 & 039

**Using this worker node to join the control node:**

sudo kubeadm join 145.100.134.38:6443 --token 8yvn73.ssi31j8g4d7knl7y --discovery-token-ca-cert-hash sha256:6fb06cc3e44f09586cfac866a4d670b5ce3d478f5888c01057e5b43e802c89fa





**Add the same kubelet arguments:**

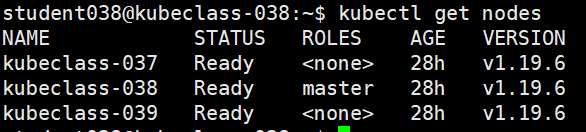
sudo echo KUBELET\_KUBEADM\_ARGS=\"--network-plugin=cni --pod-infra-container-image=k8s.gcr.io/pause:3.2 --node-ip=$(ip -4 -o a | grep -i "ens3" | cut -d ' ' -f 2,7 | cut -d '/' -f 1 | awk '{print $2}')\" | sudo tee /var/lib/kubelet/kubeadm-flags.env

**Restart the service:**

sudo systemctl restart kubelet.service

**Check the status of nodes:**

kubectl get nodes



**5.Setting up the cluster**

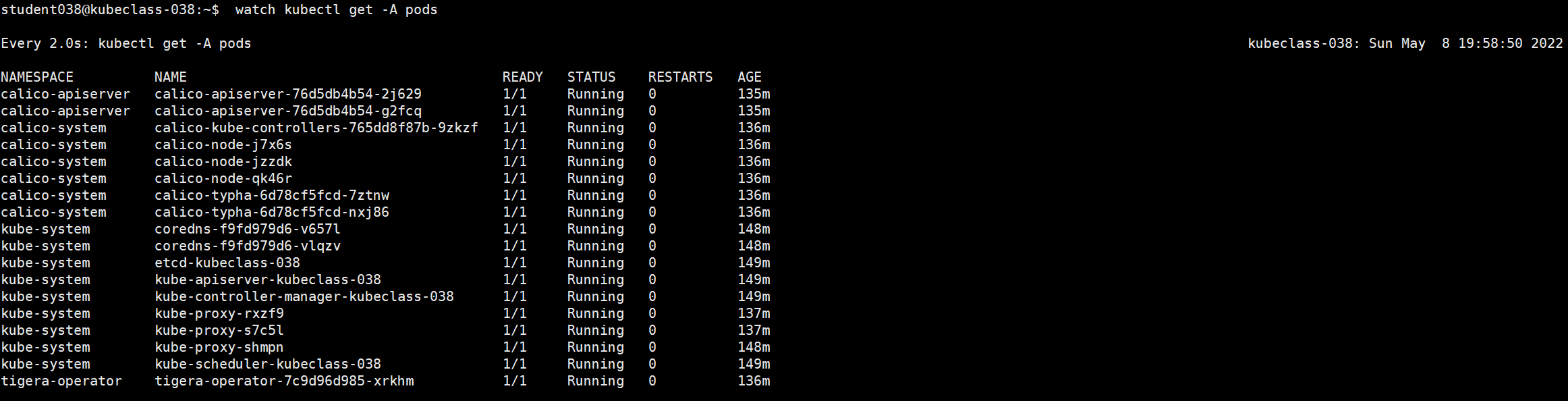
**Creates the target directory:**

mkdir -p $HOME/.kube

**Hanges the file’s owner to ourselves：**

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

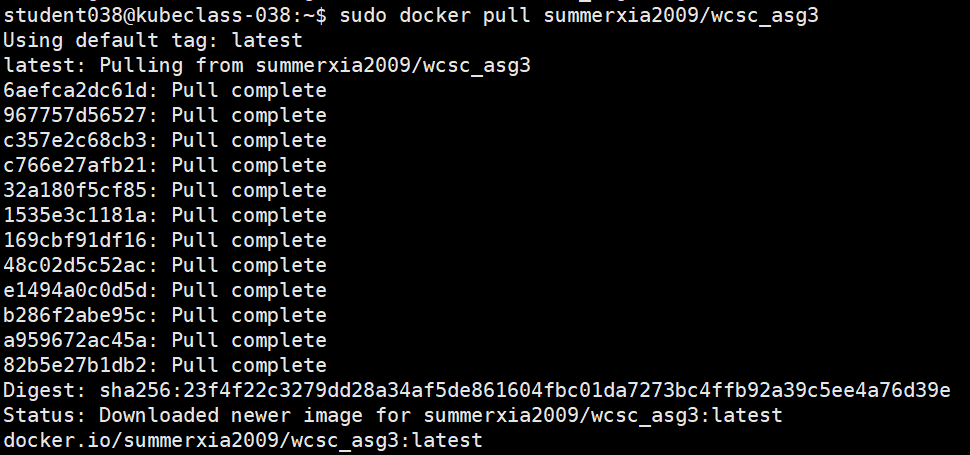
watch kubectl get -A pods



6.  **Upload the image to the server**

**The image used is from 3.1 part. It was downloaded from a public private image and pulled using the docker pull command.**

**Control node:**



**Worker node:**

After getting the image, the node will automatically go to pull.

**7. Configure the .yaml files**

vim users.yaml

vim url.yaml

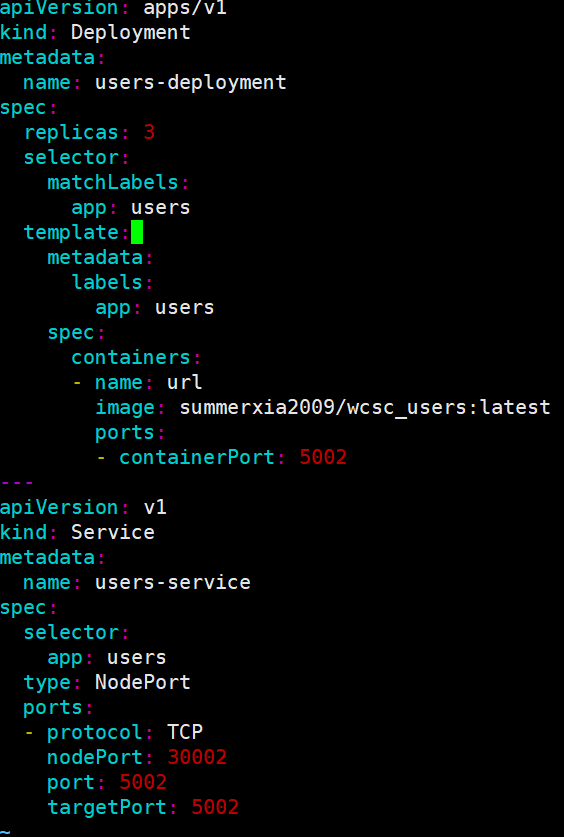
kubectl create -f url.yaml

kubectl create -f users.yaml

Implementation of the deployment and NodePort methods based on two .yaml files in the current directory

**Check the current pod status:**

kubectl get pods

If it is 6 status 1/1, then the service is successfully bui

Each .yaml file includes the deploy and service section. Each deploy comes out with three replicas, 3 pods per service. We have two services in total, so there are 6 pods.

**# Testing**

URL Shortener is tested by the port 30001 of the IP of VM-student038.

User Authentication is tested by the port 30002.

145.100.134.38:5001(/<short\_url>)

145.100.134.38:5002/users(/login)

All the containers are deployed on the master node. K8s deploys the containers to each worker node. We access both services through the IP of the master node.