**First Create a Resource Group on your Azure account and launch all the Instances in the same resource group.**

Resource Group Name = Kubernetes-HA

**SIX Ubuntu 18.04 lts** VM’s of **Standard B2s (2 vcpus, 4 GB memory)** type on your Azure account. Use the below details when creating VM’s.

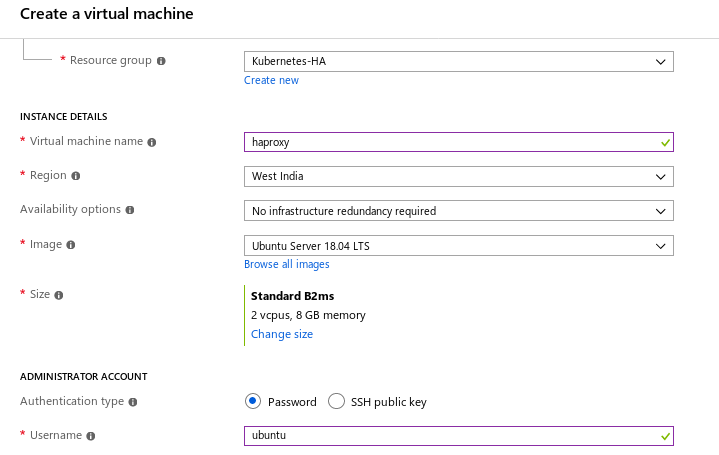
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Server Type | Host Name | username | IP Requirement and IP | Private IP’s | Public IP’s |
| HAPROXY Load-Balancer | haproxy | ubuntu | Public-IP |  |  |
| 1st-Master | master-1 | ubuntu | Public-IP |  |  |
| 2nd-Master | master-2 | ubuntu | Public-IP |  |  |
| 3rd-Master | master-3 | ubuntu | Public-IP |  |  |
| 1st-Worker | worker-1 | ubuntu | Public-IP |  |  |
| 2nd-Worker | worker-2 | ubuntu | Public-IP |  |  |

**When Creating VM’s please set username as “ubuntu” for all the Virtual Machines**

**And the hostnames as stated in the above table while creating the VM in the below steps**

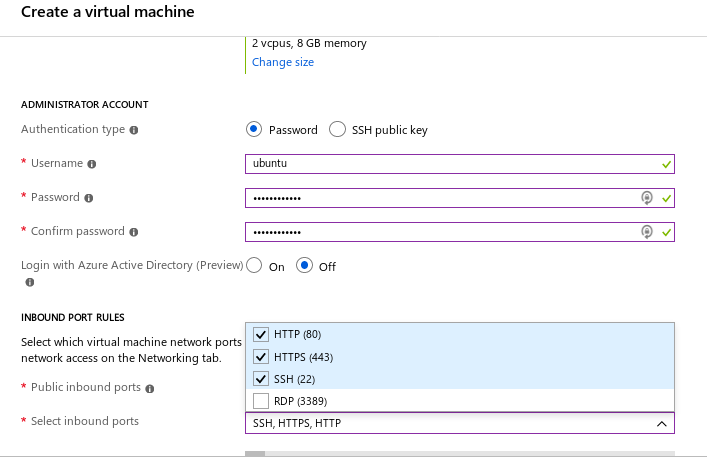
**A . Create a HAProxy LB server.**

**1.** Launch an ubuntu server 18.04 lts VM of Standard B2s (2 vcpus, 4 GB memory) type with Virtual Machine name as “haproxy” and username as “ubuntu” on your Azure account.

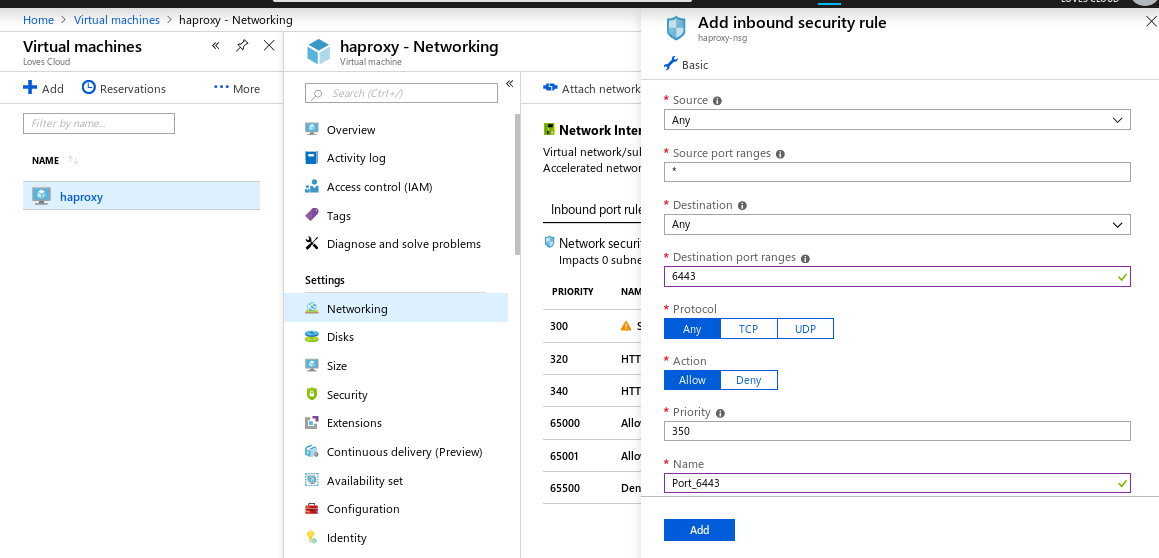


**2.** Open the below Port the below ports on the haproxy VM as shown in the screenshot.

**443, 22, 80**



**3. Open port 6443 once the VM is created as shown below.**



**4. SSH to the VM from it’s public IP.**

|  |
| --- |
| $ sudo su # apt-get update # sudo apt-get -y install haproxy # cd /etc/haproxy/  # mv haproxy.cfg haproxy.cfg.backup  # vim haproxy.cfg |

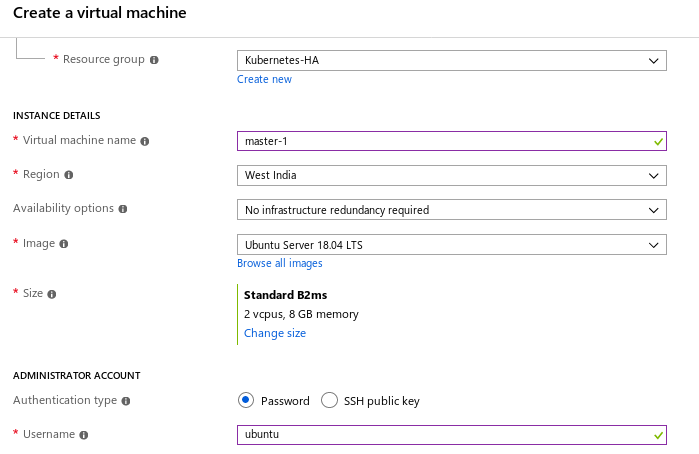
**5. Paste the below script to the haproxy.cfg**

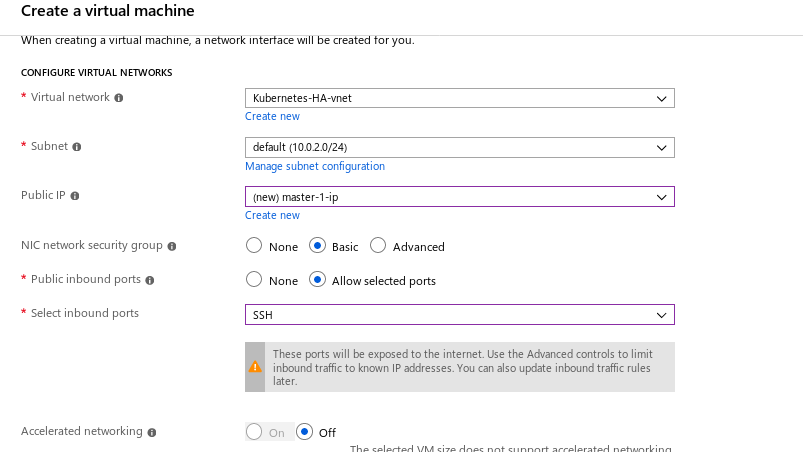
|  |
| --- |
| global  user haproxy  group haproxy  defaults  mode http  log global  retries 2  timeout connect 3000ms  timeout server 5000ms  timeout client 5000ms  frontend kubernetes  bind <ha-proxy-server-internal-ip-addr>:6443  option tcplog  mode tcp  default\_backend kubernetes-master-nodes  backend kubernetes-master-nodes  mode tcp  balance roundrobin  option tcp-check  server master-1 <internal-ip-first-master>:6443 check fall 3 rise 2 # Update the hostname   server master-2 <internal-ip-second-master>:6443 check fall 3 rise 2 # Update the hostname  server master-3 <internal-ip-third-master>:6443 check fall 3 rise 2 # Update the hostname |

Save and exit by pressing the **esc key** then **wq** andpress **enter** to exit.

**B . Create the first Master Node**

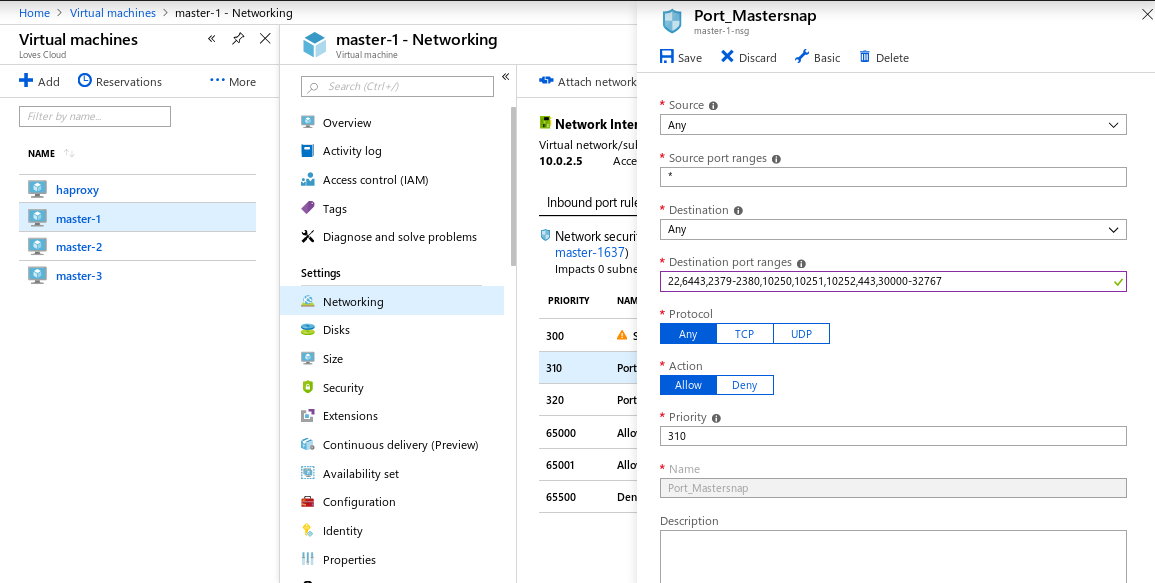
**1.** Launch an ubuntu server 18.04 lts VM of Standard B2s (2 vcpus, 4 GB memory) type with username as ubuntu on your Azure account.





**2.** Once the VM is created, go to the Networking Section and open the Ports as shown in the below screenshot.

Ports : **22,6443,2379-2380,10250,10251,10252,443,30000-32767**



**3.** SSH to the **master-1** VM from its Public-IP.

|  |
| --- |
| # ssh ubuntu@<Public-IP-master-1> |

## **Installing kubeadm, kubelet, kubectl and Docker (runtime)**

**4.** Run the below commands as a privileged user.

|  |
| --- |
| $ sudo su # apt-get update |

**5**. Now run below command to add Kubernetes certificates.

|  |
| --- |
| # apt-get install -y apt-transport-https curl # 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 https://apt.kubernetes.io/ kubernetes-xenial main EOF # apt-get update  # apt-get install \  apt-transport-https \  ca-certificates \  curl \  gnupg2 \  software-properties-common -y |
|  |

**6**. Install kubectl, kubeadm, kubelet and CNI

|  |
| --- |
| # apt-get install kubectl=1.13.4-00 kubeadm=1.13.4-00 kubelet=1.13.4-00 kubernetes-cni=0.6.0-00  # apt-mark hold kubelet kubeadm kubectl |

**7.** Install Docker .

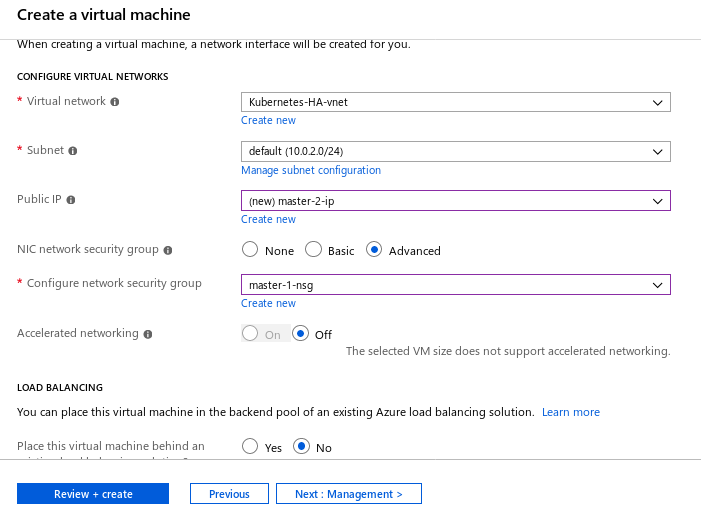
|  |
| --- |
| # curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -  # apt-key fingerprint 0EBFCD88  # add-apt-repository \  "deb [arch=amd64] https://download.docker.com/linux/ubuntu \  $(lsb\_release -cs) \  stable" # apt-get update # sudo apt-get install docker-ce=18.06.1~ce~3-0~ubuntu -y  # apt-mark hold docker-ce |

Follow the steps ( 3,4,5,6,7) on all the master node’s.

**C . Create the Second and third Master Node**

**1.** Launch two ubuntu 18.04 lts VM of Standard B2s (2 vcpus, 4 GB memory) type on your Azure account with username as ubuntu.

**2.** While creating the **master-2** and **master-3 VM’s**, under the Networking section select the Advanced option for the **NIC network security group > Configure network security group** and select the security group for these masters of the **master-1-nsg**, as shown in the below image.



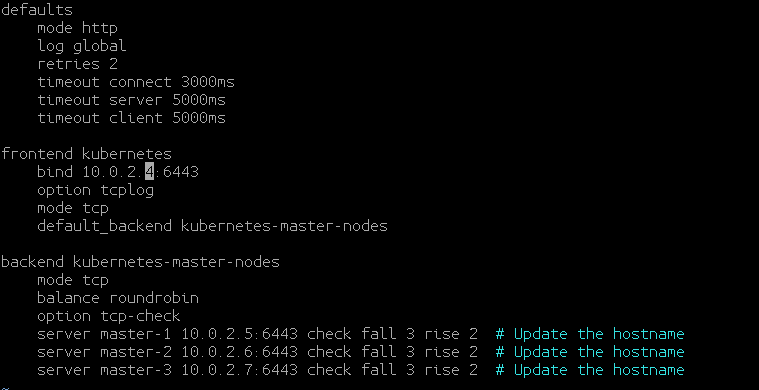
**3.** SSH to the master-2 and master-3 VM and repeat the steps (3,4,5,6,7)from **Part** **B**.to install kubectl, kubeadm, kubelet and docker.

**D . Setting up Stacked control plane**

**1.** SSH to the **HAProxy VM** and run the below command.

|  |
| --- |
| $ sudo su # vim /etc/haproxy/haproxy.cfg |

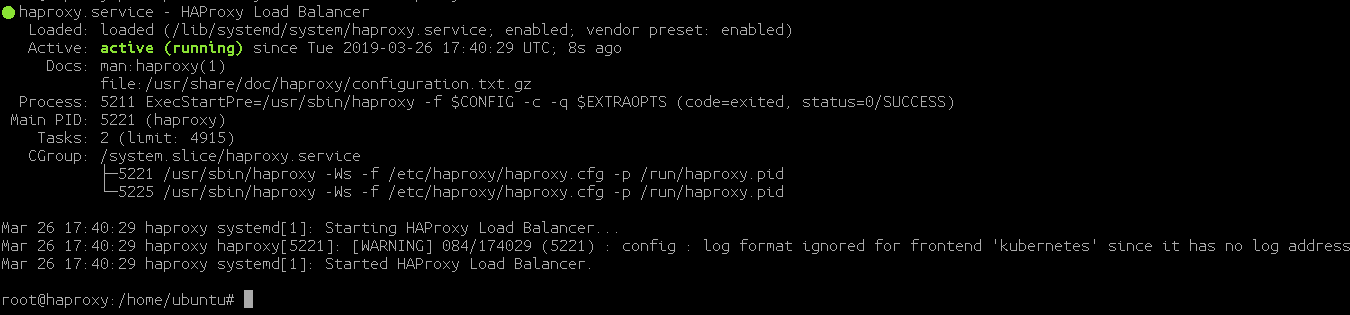
**2.** Update **<ha-proxy-server-internal-ip-addr>, <internal-ip-first-master>, <internal-ip-second-master> and <internal-ip-third-master>** with the internal ip of the haproxy and masters respectively as shown in the below image.



**Save and exit by the escape key and :wq ( hit enter to exit).**

**3.** Restart the **Haproxy** service

|  |
| --- |
| # systemctl restart haproxy # systemctl status haproxy |



**E. Initiating the Cluster**

**1.** SSH to the **First Maste**r Node from the HAProxy VM and run sudo su to become privileged user.

|  |
| --- |
| # ssh ubuntu@<public-IP-master-1> |

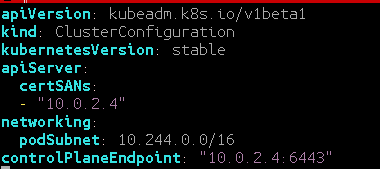
|  |
| --- |
| $ sudo su # vim kubeadm-config.yaml |

**2.** Paste the below content

Update “ha-proxy-server-internal-ip-addr” in the below script

|  |
| --- |
| apiVersion: kubeadm.k8s.io/v1beta1 kind: ClusterConfiguration kubernetesVersion: stable apiServer:  certSANs:  - "ha-proxy-server-internal-ip-addr" networking:  podSubnet: 10.244.0.0/16 controlPlaneEndpoint: "ha-proxy-server-internal-ip-addr:6443" |

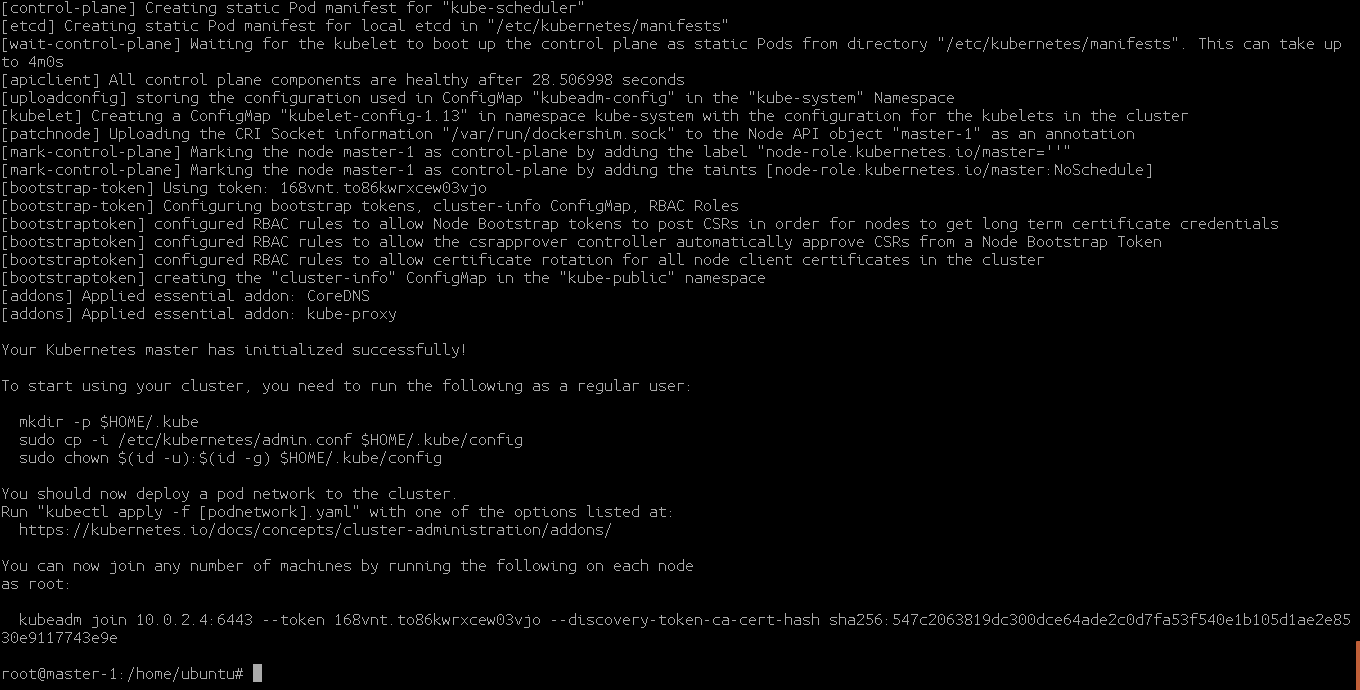
**Save and exit by the escape key and :wq ( hit enter to exit).**



**3.** Now, Create the cluster with the below command.

|  |
| --- |
| # sudo kubeadm init --config=kubeadm-config.yaml |

**4. Save the Join command output and save it for further reference.**

****

**5. Once the cluster is up run the below commands as shown in the output.**

|  |
| --- |
| # mkdir -p $HOME/.kube  # sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config # sudo chown $(id -u):$(id -g) $HOME/.kube/config  # cp ~/.kube/config /home/ubuntu |

**6.** Press **Ctrl+d**

|  |
| --- |
| $ mkdir ~/.kube/ $ sudo cp /home/ubuntu/config ~/.kube/  $ sudo chown $(id -u):$(id -g) $HOME/.kube/config  $ kubectl get nodes |

**7.** Install the CNI

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

**8. Copy the certificate**

On the other control-planes ( master-2,3…)

Copy the certificate files from the first control plane node (First Master) to the rest:

Update the username if required.

Update the **“MASTER\_2\_PRIVATE\_IP”** in the below script

|  |
| --- |
| **USER=ubuntu # customizable**  **CONTROL\_PLANE\_IPS="<MASTER\_2\_PRIVATE\_IP>"**  **for host in ${CONTROL\_PLANE\_IPS}; do**  **scp /etc/kubernetes/pki/ca.crt "${USER}"@$host:**  **scp /etc/kubernetes/pki/ca.key "${USER}"@$host:**  **scp /etc/kubernetes/pki/sa.key "${USER}"@$host:**  **scp /etc/kubernetes/pki/sa.pub "${USER}"@$host:**  **scp /etc/kubernetes/pki/front-proxy-ca.crt "${USER}"@$host:**  **scp /etc/kubernetes/pki/front-proxy-ca.key "${USER}"@$host:**  **scp /etc/kubernetes/pki/etcd/ca.crt "${USER}"@$host:etcd-ca.crt**  **scp /etc/kubernetes/pki/etcd/ca.key "${USER}"@$host:etcd-ca.key**  **scp /etc/kubernetes/admin.conf "${USER}"@$host:**  **done** |

Update the **“MASTER\_3\_PRIVATE\_IP”** in the below script

|  |
| --- |
| **USER=ubuntu # customizable**  **CONTROL\_PLANE\_IPS="<MASTER\_3\_PRIVATE\_IP>"**  **for host in ${CONTROL\_PLANE\_IPS}; do**  **scp /etc/kubernetes/pki/ca.crt "${USER}"@$host:**  **scp /etc/kubernetes/pki/ca.key "${USER}"@$host:**  **scp /etc/kubernetes/pki/sa.key "${USER}"@$host:**  **scp /etc/kubernetes/pki/sa.pub "${USER}"@$host:**  **scp /etc/kubernetes/pki/front-proxy-ca.crt "${USER}"@$host:**  **scp /etc/kubernetes/pki/front-proxy-ca.key "${USER}"@$host:**  **scp /etc/kubernetes/pki/etcd/ca.crt "${USER}"@$host:etcd-ca.crt**  **scp /etc/kubernetes/pki/etcd/ca.key "${USER}"@$host:etcd-ca.key**  **scp /etc/kubernetes/admin.conf "${USER}"@$host:**  **done** |

CONTROL\_PLANE\_IPS="10.0.0.7 10.0.0.8" in this example ( Internal IP address of master-2 and master-3)

**Caution:** Copy only the certificates in the above list. kubeadm will take care of generating the rest of the certificates with the required SANs for the joining control-plane instances. If you copy all the certificates by mistake, the creation of additional nodes could fail due to a lack of required SANs

**9. SSH** to **master-2** and **master-3** and move the files created by the previous step where scp was used:

|  |
| --- |
| # ssh ubuntu@<master-2-public-ip> $ sudo su root@master-2:/home/ubuntu# ls admin.conf ca.crt ca.key etcd-ca.crt etcd-ca.key front-proxy-ca.crt front-proxy-ca.key master.sh sa.key sa.pub |

**10.** Run the below command to copy the certs to /etc/kubernetes/pki/ on **master-2** and **master-3**

|  |
| --- |
| USER=ubuntu # customizable mkdir -p /etc/kubernetes/pki/etcd cp /home/${USER}/ca.crt /etc/kubernetes/pki/ cp /home/${USER}/ca.key /etc/kubernetes/pki/ cp /home/${USER}/sa.pub /etc/kubernetes/pki/ cp /home/${USER}/sa.key /etc/kubernetes/pki/ cp /home/${USER}/front-proxy-ca.crt /etc/kubernetes/pki/ cp /home/${USER}/front-proxy-ca.key /etc/kubernetes/pki/ cp /home/${USER}/etcd-ca.crt /etc/kubernetes/pki/etcd/ca.crt cp /home/${USER}/etcd-ca.key /etc/kubernetes/pki/etcd/ca.key cp /home/${USER}/admin.conf /etc/kubernetes/admin.conf |

**11.** Start kubeadm join on this node using the join command that was previously given to you by kubeadm init on the first Master-1. It should look something like this:

Example Token run command on Master-2

|  |
| --- |
| sudo kubeadm join <HA-Proxy-Internal-IP>:6443 --token j04n3m.octy8zely83cy2ts --discovery-token-ca-cert-hash sha256:84938d2a22203a8e56a787ec0c6ddad7bc7dbd52ebabc62fd5f4dbea72b14d1f --experimental-control-plane |

Run the token on the Master-2 and Master-3 with “TOKEN” followed by --experimental-control-plane as shown above.

**12.** Type the following on the **MASTER-1 VM**  and watch the pods of the components get started:

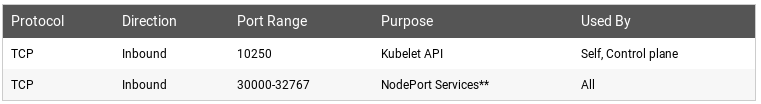
|  |
| --- |
| kubectl get pod -n kube-system -o wide |

**13.** Repeat steps 9-12 on **NODE master-3**

**F . Setting up the Worker VM’s**

**1.** Launch two ubuntu server 18.04 lts VM of Standard B2s (2 vcpus, 4 GB memory) type with username as ubuntu and virtual machine name as **worker-1** and **worker-2** on your Azure account.

**2.** Once the VM is created, go to the Networking Section and open the Ports as shown in the below screenshot.



**3.** SSH to the worker-1 VM from the Public IP

|  |
| --- |
| $ sudo su # apt-get update && apt-get install -y apt-transport-https curl |

**4.** Follow the below commands to **Install Docker kubeadm , kubelet and kubernetes-cni**

|  |
| --- |
| # 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 https://apt.kubernetes.io/ kubernetes-xenial main EOF # apt-get update  # apt-get install \  apt-transport-https \  ca-certificates \  curl \  gnupg2 \  software-properties-common -y |

**5. Install Docker kubeadm , kubelet and kubernetes-cni**

|  |
| --- |
| # curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add - # apt-key fingerprint 0EBFCD88  # add-apt-repository \  "deb [arch=amd64] https://download.docker.com/linux/ubuntu \  $(lsb\_release -cs) \  stable" # apt-get update  # sudo apt-get install docker-ce=18.06.1~ce~3-0~ubuntu -y |
| # apt-mark hold docker-ce |

|  |
| --- |
| # sudo apt-get install kubeadm=1.13.4-00 kubelet=1.13.4-00 kubernetes-cni=0.6.0-00 -y  # apt-mark hold kubelet kubeadm |

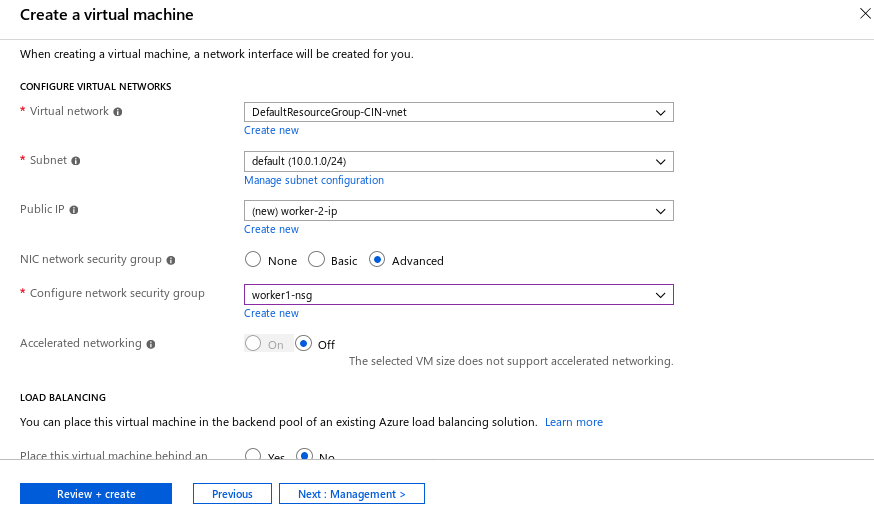
**6.** Once kubelet kubeadm docker-ce is installed run the join-token command from the master-1 during the initial cluster setup **without** the **--experimental-control-plane** to join this VM as a worker VM

**Example:**

|  |
| --- |
| kubeadm join <HA-Proxy-Internal-IP> --token 96vgsk.j9hakuek66zl6m1a --discovery-token-ca-cert-hash sha256:3ab1878ace91c113e642e8dc45b62fc651e6d731a7bfa0bdf360cdd5ff2a3621 |

**7.** Repeat **steps 1-6** and skip **step 2** on the **worker-2** VM to join the cluster.

Note- When creating the **worker-2 VM** select the **NIC network Security group from advanced** of **worker-1**



**F . Setting up the HAProxy VM as the Workstation**

SSH to the **HAProxy** VM.

|  |
| --- |
| $ sudo su |

Run the below commands.

|  |
| --- |
| # apt-get update && apt-get install -y apt-transport-https curl # 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 https://apt.kubernetes.io/ kubernetes-xenial main EOF  # apt-get update  # sudo apt-get install kubectl=1.13.4-00 -y # apt-mark hold kubectl |

Now, SSH to the **master-1 VM**

|  |
| --- |
| $ sudo su # scp ~/.kube/config ubuntu@<internal-ip-haproxy>:/home/ubuntu |

Once the file is copied, SSH back to the **HAProxy VM** RUN the below command on the **HAProxy VM**.

|  |
| --- |
| # mkdir ~/.kube/  # cp /home/ubuntu/config ~/.kube/  # kubectl get nodes |