

Cluster Kuberentes Local

Requisitos:

- Minimo de 8GB Ram (Dica para um aprendizado mais eficiente 16GB)
- VS Code(não obrigatório)
- VirtualBox
- Vagrant (Uso dele é para subir a infra, mas não é obrigatório)
- Lens (IDE para uma familiarização gráfica, não abordarei ele a fundo para melhor estudos indico o uso da documentação “docs.k8slens.dev/main/”)

Considerações a fazer:

O SO hospedeiro que usarei será linux, mas se quiser utilizar outro não haverá problemas, por heterogeneidade o VirtualBox será utilizado.

Sinopse:

Neste laboratório abordaremos a criação de um cluster k8s local onde teremos 3 ubuntu 18.04, uma control-plane e duas workers, vamos implantar micro-serviços, explorar comandos e abordaremos o uso basico de funcionalidades que o Kubernetes tem a oferece para melhor gerenciamento.

Inicialização do laboratório:

1. Presumindo que você já esta com os programas que abordamos em **Requisitos** instalados, vamos abrir o vagrantfile no VSCode e realize as alterações dos IPs das maquinas para o ip da sua rede.

```
config.vm.define "vm2" do |vm2|  
  vm2.vm.network "public_network", ip: "192.168.10.127"
```

No meu caso o range de IP aqui é 192.168.10.0

2. Agora abra o **master.sh** localize e altere o IP do `--apiserver-advertise-address=` para o mesmo da vm “master” que será nossa control-plane.

```
sudo kubeadm init --pod-network-cidr=10.244.0.0/16 --apiserver-advertise-address=192.168.10.126
```

3. Abro o próprio terminal do VSCode e execute o comando => `vagrant up master` ele irá iniciar a vm master e dará a opção de interface de rede, no caso aqui usarei a WI-FI então escolherei a 1 como brigde.

```
Bringing machine 'master' up with 'virtualbox' provider...  
==> master: Importing base box 'ubuntu/bionic64'...  
==> master: Matching MAC address for NAT networking...  
==> master: Checking if box 'ubuntu/bionic64' version '20220107.0.'  
to date...  
==> master: Setting the name of the VM: ProjetoK8S_master_16549015  
5  
==> master: Fixed port collision for 22 => 2222. Now on port 2200.  
==> master: Clearing any previously set network interfaces...  
==> master: Available bridged network interfaces:  
1) wlp0s20f3  
2) eno1  
3) docker0  
==> master: When choosing an interface, it is usually the one that  
==> master: being used to connect to the internet.  
==> master:  
master: Which interface should the network bridge to? 1
```

4. Ele vai começar a instalar as dependências do tudo que precisamos para criar nosso cluster kubernetes assim que ele finalizar vamos voltar ao vagrantfile para realizarmos alterações de memória das vms workers, devemos comentar onde está declarado na vm master as configurações de memória e CPU.

```
# MASTER será a Control-plane
config.vm.define "master" do |master|
#   config.vm.provider "virtualbox" do |v|
#       v.memory = 2048
#       v.cpus = 2
#   end
end
master.vm.hostname = "master"
```

5. Salve, e no terminal do VSCode execute o comando => `vagrant up worker1 worker2`

```
er2
Bringing machine 'worker1' up with 'virtualbox' provider...
Bringing machine 'worker2' up with 'virtualbox' provider...
==> worker1: Importing base box 'ubuntu/bionic64'...
```

Siga o mesmo processo de escolha da interface que fizemos na master.

Nota:(se sua máquina for 8GB de RAM execute apenas uma `vagrant up worker1`).

6. Assim de instalarem acesse as workers como o comando => `vagrant ssh worker1`

```
~/Documentos/ProjetoK8S$ vagrant ssh worker1
```

(para a worker2 é só substituir o nome)

7. Agora vá até o diretório `cd /vagrant` verifique se existe um executável chamado `join.sh`

```
vagrant@worker1:~$ cd /vagrant
vagrant@worker1:/vagrant$ ls
Documentação.odt  join.sh  master.sh  ubuntu-bionic-18.04-cloudimg-console.log  vagrantfile  worker.sh
```

execute o `join` ele retornará a imagem

```
vagrant@worker1:/vagrant$ sudo sh join.sh
[preflight] Running pre-flight checks
[preflight] Reading configuration from the cluster...
[preflight] FYI: You can look at this config file with 'kubectl -n kube-system get cm kubeadm-config -o yaml'
[kubelet-start] Writing kubelet configuration to file "/var/lib/kubelet/config.yaml"
[kubelet-start] Writing kubelet environment file with flags to file "/var/lib/kubelet/kubeadm-flags.env"
[kubelet-start] Starting the kubelet
[kubelet-start] Waiting for the kubelet to perform the TLS Bootstrap...

This node has joined the cluster:
* Certificate signing request was sent to apiserver and a response was received.
* The Kubelet was informed of the new secure connection details.

Run 'kubectl get nodes' on the control-plane to see this node join the cluster.
```

Nota: repita o mesmo processo 6 e 7 para a worker2.

8. Para sair da worker é só executar `exit`

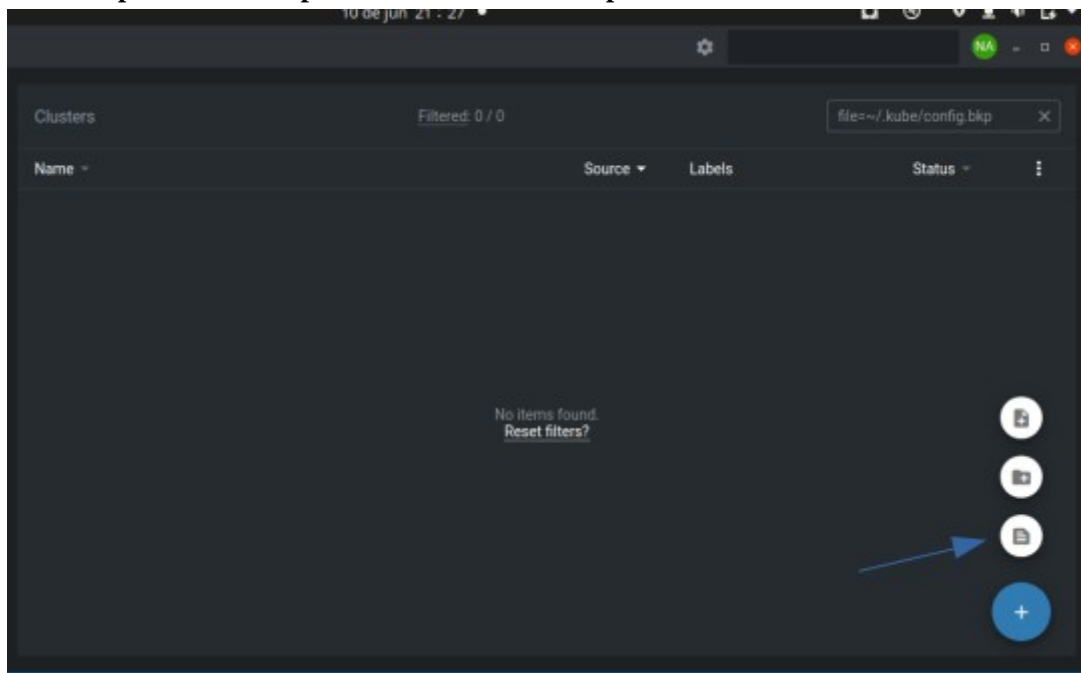
```
vagrant@worker1:/vagrant$ exit
logout
Connection to 127.0.0.1 closed.
```

Comemore você configurou seu cluster Kubernetes!!!

9. Aceda a vm master com o comando do passo 6, vamos pegar o arquivo de configuração para executar o `kubectl` na IDE LENS, execute “`cd .kube`” e abra arquivo “`config`”. Com o mouse selecione tudo o que ele exibiu como na imagem abaixo. Nota: a imagem está cortada mas você deve selecionar tudo que ele está exibindo.

```
vagrant@master:~$ cd .kube
vagrant@master:~/.kube$ cat config
apiVersion: v1
clusters:
- cluster:
    certificate-authority-data: LS0tLS1CRUdJTiBDRVJUSUZJQ0FURSB0tLS0tCk1JSUMvakNDQWVhZG
    F3SUJBZ0lCQURBTkNa3Foa2lH0Xcw0kFRc0ZBREFTWVJNd0VRWURWUWFERXdwcmRXSmwKY201bGRHVnpNQjRY
    RFRJeU1EWXhNREL6TVRBek9Wb1hEVE15TURZd056SxpNVEF6TlZvd0ZURVRNqkVHQTFRQpBeE1LYTNwaVpYSr
    VawFJ3Y3pDQ0FTSXdEUVlKS29aSWh2Y05BUUVCQlFBRGdnRVBIBRENDQVFvQ2dnRUJBS2IxY3amd0VlRzd0YU
    NEdLUXFJQUdteFZpYlB1RndmRS9oTmPyK3JUtk1ZZXBxL3J1QUFlbWtDRmJwZnRsNzVoV2sKbDl4ODJlIbTg1SH
    9qYUdLZDBJdnk0Y2V4clEwYjNncDBzTGl2SFN0UTE5YlVJZXBKMEZiSVVKcU5IUmF0ZEt2awo4M21xdXE1NXBp
    bmFNT091VURrbUkwTDFsS0J6NEF6UWhBZHE4Y3l2VFRJbWwNa3VJRWxH0WJERmtXTWxmcnNlCnFnVzdISkdCUH
    Z0aGlia05Za1E4M0tnSudvbkpZU2w5b1B0R1A0YW1XlgrUEptRz14QVQxdVJvYw0ajl2dUQKSjRTR2UydVp6
    VTcxThHdGk5cFpJUVNVmWliVW9kwUfKS3BqN1BVclwVjZ3TXU3dktIVkMvRzZxMzFLQXJaMQpMSkI3YnB6L6
    YrTGdQWxVXU1ZjQ0F3RUFBYU5aTUZjd0RnWURWUjBQOVFILOjBUURBZ0trTUE4R0ExVWRFd0VCCi93UUZNQU1Q
    QWY4d0hRWURWUjBPQkJZRUZERzhoTldHTlZhU2RzRFZKTzgxS2JIZTU4VnNNQlVHQTFRVZEVUU8KTUF5Q0NtdD
    FZbVZ5Ym1WMFpYTXdEUVlKS29aSWh2Y05BUUVMQlFBRGdnRUJBQ0Y3c3JoM1BNbVJKVndoMSs5bwpHM1h6ZWtv
    MwZyazZPYks2Q0x2Wk9VUEM3QWZVY3Ntr3l6SmtPVGJYXkZrY0xCajNYOUJmY2svWmNGWC9VnkZrClJlY2RTMT
    VDbGFKMUp5eG5MSU1TVGRXcnExamdZTXNNR0ZlTkpnYzV5RUR6ekNTM1pkS1cySWV6eEF5NVZaeVYKb3VEbFMy
    a2JaSlzSlZaeUI4WWhjQkNnYW1UMk00Y2lhVmg4SzdqbGRtaGJkYWpDTkttSEd3dzVEbHkvWXMzQWp0aCtNND
    QvWktlbG1mNlV5NmVGSUpwZTQwcGwxR24zcjJ0R0dBRTYxRzRoSmZ0QXl5M0FGKzVEclXZVpLV2UzCkMzaUk3
    RUDVdUplRVVoL3ZTYktkNy8vb0l5azNFdERPZkgvOFdGalVJUmhoWEEzTzdXL3hPY1U4NHJlMmwrR08KaTk0PQ
    otLS0tLUVORCBDRVJUSUZJQ0FURSB0tLS0tCg==
    server: https://192.168.10.127:6443
  name: kubernetes
contexts:
- context:
```

10. Após você ter copiado o abra seu LENS passe a seta no +




Selecione o file e cole dentro da caixa o arquivo copiado igual na imagem abaixo

Clusters added here are **not** merged into the `~/.kube/config` file. [Read more about adding clusters.](#)

```
1  apiVersion: v1
2  clusters:
3  - cluster:
4    |   certificate-authority-data: LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0tC
5    |   server: https://192.168.10.127:6443
6    |   name: kubernetes
7  contexts:
8  - context:
9    |   cluster: kubernetes
10   |   user: kubernetes-admin
11   |   name: kubernetes-admin@kubernetes
12  current-context: kubernetes-admin@kubernetes
13  kind: Config
14  preferences: {}
15  users:
16  - name: kubernetes-admin
17    |   user:
18    |   client-certificate-data: LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0tCk1J
19    |   client-key-data: LS0tLS1CRUdJTiB0EgUUFJJVJkFURSBLRVktLS0tLQpNSU1F
```

Role para baixo e clique em “add cluster”, você vai ver que ele foi adicionado

Clusters		1 item		Search...	
Name ▾	Source ▾	Labels	Status ▾		
 kubernetes-admin@kubernetes	local		connected		

Clique nele para conectar ao cluster.

Agora vamos brincar um pouco!!!