

Cluster Kubernetes Local

Requisitos:

- Mínimo de 8GB RAM (recomendável: 16GB)
- VS Code (não obrigatório)
- VirtualBox
- Vagrant (o uso dele é para subir a infra, mas não é obrigatório)
- Lens (IDE para uma familiarização gráfica, não será abordado a fundo para melhor estudos indico o uso da documentação “docs.k8slens.dev/main/”)

Considerações:

O SO hospedeiro utilizado nesse tutorial é o Linux, porém pode ser utilizado qualquer outro de sua preferência. Por heterogeneidade, o Hypervisor utilizado será o VirtualBox.

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 básico de funcionalidades que o Kubernetes tem a oferecer para melhor gerenciamento.

Inicialização do laboratório:

1. Presumindo que você já esteja com os programas que abordamos em **Requisitos** instalados, vamos abrir o vagrantfile no VSCode e realize as alterações dos IPs das máquinas 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. Abra 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 bridge:

```
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 de tudo que precisamos para criar nosso cluster Kubernetes assim que ele finalizar vamos voltar ao vagrantfile para realizarmos alterações de memória das VM's 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'...
1
```

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
15 (CPU: 1, 15.0, 166, memoria: 4096, 64)
```

(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.sh` 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 apiserer 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`

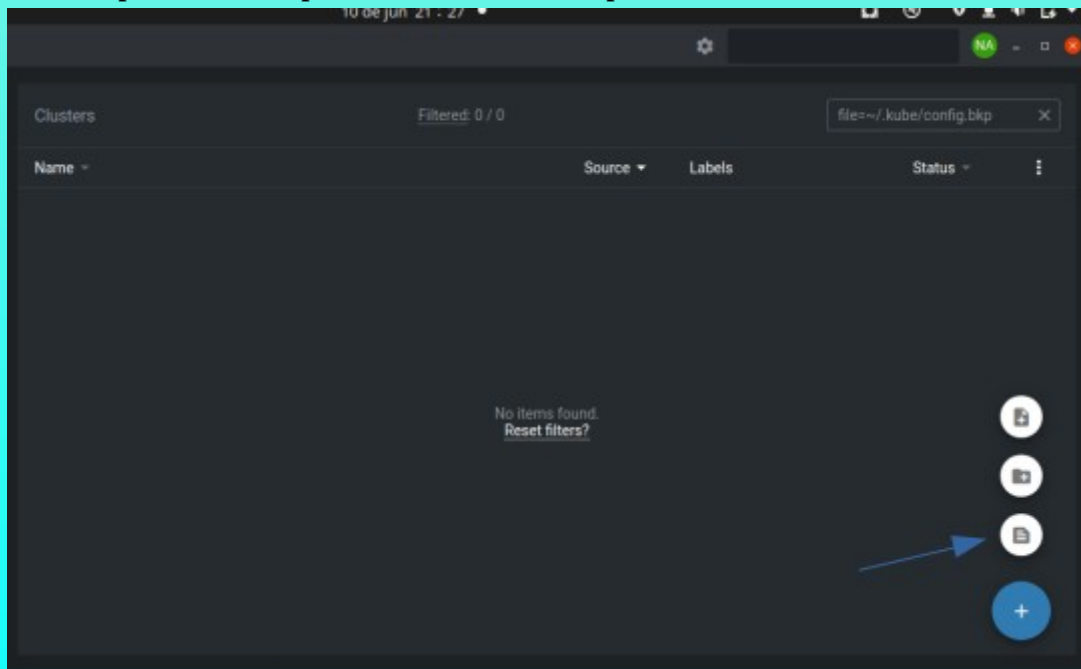
```
vagrant@worker1:/vagrant$ exit
logout
Connection to 127.0.0.1 closed.
```

Comemore você configurou seu cluster Kubernetes!!!

9. Acesse 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: LS0tLS1CRUdJTiBDRVJUSUZJQ0FUR50tLS0tCk1JSUMvakNDQWVhZG
    F3SUJBZ0lCQURBTkNa3Foa2lH0XcwQkFRc0ZBREFTVjN0d0VRWURWUWFERXdwcmRXSmwKY201bGRHVnpNQjRY
    RFRJeU1EWXhNREl6TVRBek9wb1hEVE15TURZd056SxpNVEF6T1Zvd0ZURVRNQkVHQTFVRQpBeE1LYTNWVpYSR
    VawFJys3PDQ0FTSXdEUVlKS29aSWh2Y05BUUVCQlFBRGdnRVB0R0tRUE40d0Y3amR0b0VlRzd0YU
    NEdLUXFJQUdteFZpYlB1RndmRS9oTmPyK3JUtk1ZZXBxL3J1QUFlbWtDRmJwZnRsNzVoV2sKbDl4ODJibTg1SH
    9qYUdLZDBJdnk0Y2V4clEwYjNncDBzTGl2SFN0UTE5YlVJZXBKMEZiSVVKcU5IUmF0ZEt2awo4M21xdXE1NXBp
    bmFNT091VURrbUkwTDFsS0J6NEF6UWhBZHE4Y3l2VFRJbWwNa3VJRWxHQWJERmtXTWxmcnNlCnFnVzdISkdCUH
    Z0aGlia05Za1E4M0tnSudvbkpZU2w5b1B0R1A0YWlXlgrUEptRz14QVQxdVJvYWk0aj12dUQKSjRTR2UydVp6
    VTCxbThHdGk5cFpJUUVNMW1iVW9kWUFS3BqN1Bvc1lwVjZ3TXU3dktIVkMvRzZxMzFLQXJaMQpMSkI3YnB6L6
    YrTgDQWVXU1ZjQ0F3RUFBU5aTUZjd0RnWURWUjBQOVFILOjBUURBZ0trTUE4R0ExVWRf0VCCi93UUZNU1Q
    QWY4d0hRWURWUjBPQkJZRUZERzhoTldHTlZhU2RzRFZKTzgxS2JIZTU4VnNNQlVHQTfVZEVURUU8KTUF5Q0NtdD
    FZbVZ5Ym1wMFpYTXdEUVlKS29aSWh2Y05BUUVMQlFBRGdnRUJBQ0Y3c3JoM1BNbVJKVndoMS55bWpHM1h6ZWtV
    MwZyazZPYks2Q0x2Wk9VUEM3QWZVY3Ntr3l6SmtPVGJXYkZrY0xCajNYOUJmY2svWmNGWC9VnkZrClJlY2RTMT
    VDBGFKMUp5eG5MSU1TVGRXcnExamdZTXNNR0ZlTkpYzV5RUR6ekNTM1pkS1cySWV6eEF5NVZaeVYKb3VEbFMy
    a2JaSlzSlZaeUI4WWhjQkNnYW1UMkQ0Y2lhVmg4SzdqbGRtaGJkYWpDTkttSEd3dzVEbHkvWXZmQWp0aCtNND
    QvWktlbG1mNlV5NmVGSUpwZTQwcGwxR24zcj0R0dBRtYxRzRoSmZ0QXl5M0FGKzVEc1dXZVpLV2UzCkMzaUk3
    RUdVdUplRVVoL3ZTYktkNy8vb0l5azNFdERPZkgvOFdGalVJUmhoWEEzTzdXL3hPY1U4NHJlMwR08KaTk0PC
    otLS0tLUVORCBDRVJUSUZJQ0FUR50tLS0tCg==
    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.

✕

ESC

```
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: LS0tLS1CRUdJTiBSU0EgUFJJVkFURSBLRVktLS0tLQpNSU1F
19         client-key-data: LS0tLS1CRUdJTiBSU0EgUFJJVkFURSBLRVktLS0tLQpNSU1F
```

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!!!