

## PROJETO COMPASS

Eduardo Luan da Rosa – PUCPR

Gabriel Bicalho Ferreira - UFOP

- **Primeiros passos:**

O minikube é uma implementação do kubernetes que cria uma máquina virtual na máquina local e implanta um cluster simples contendo apenas um nó. Por padrão, o comando minikube start, aloca 2200 MB de memória RAM, logo, a VM oracle Linux 8.5 necessitaria de no mínimo 3GB de RAM para iniciar o minikube.

Ao tentar criar uma Máquina Virtual com 3GB RAM no computador do Eduardo se deparamos com um erro chamado “VirtualBox – Guru Meditation” no que consistia em um uso excessivo de memória deste computador. A solução foi criar uma VM com menos memória e quando foi criada aumentamos a quantidade de memória, porém, ao executar o comando “minikube start” o erro voltou a aparecer. Devido a esse problema, foi pensado em fazer apenas em uma máquina, no caso, a do Gabriel.

Ao tentar esta solução, foi possível criar uma VM com os 3GB RAM e instalar o minikube e até mesmo rodar o comando “minikube start”, entretanto, ao executá-lo começamos a ter instabilidade no sistema, o que chegava a ocasionar timeout (tempo excedido), e muitas vezes o travamento da máquina no geral. Sendo necessário a reinicialização do computador.

Apresentado o problema para os instrutores, foi dada a solução de avançar para os passos que tínhamos que fazer no Jenkins, com o propósito de que nenhum de nós fossemos prejudicados por falta de recurso.

Entretanto, foi possível contornar este problema. E quando a inconsistência era existente a forma de resolver isso foi fazer SNAPSHOTS.

- **Preparando o ambiente:**

- Foi utilizado o VirtualBox com a ISO Oracle Linux 8.5;
- Com 3GB RAM; 2 CPU; 30GB HD;
- Placa em modo Bridge.

- **Fixando o IP:**

Para fixar o IP foi necessário editar o arquivo **ifconf-enp0s3**. Está localizado no diretório **/etc/sysconfig/system-network**.

- **Instalação das ferramentas:**

1. **Docker:**

Para instalar o Docker foi seguido o tutorial do site oficial da oracle-base, disponível em <https://oracle-base.com/articles/linux/docker-install-docker-on-oracle-linux-ol8>. Foi gerado um script com os comandos e executado na VM.

2. **Minikube:**

A instalação foi feita seguindo as instruções do site oficial do minikube, que se encontra no site: <https://minikube.sigs.k8s.io/docs/start/>

Foi gerado um script para fazer o download do arquivo .rpm e instalar. Logo após, foi executado o comando minikube start.

3. **Jenkins:**

O Jenkins foi instalado como um Pod do minikube. Para isso, foi utilizado dois arquivos .yaml, um referente ao service e outro referente ao deployment do Jenkins. No repositório do GIT, contém os arquivos e a forma de executá-los.

O tutorial utilizado, é a segunda opção de instalação oferecida no site:

<https://www.jenkins.io/doc/book/installing/kubernetes/>

Os arquivos .yaml foram retirados desse site.

Foi necessário fazer algumas pequenas alterações nos arquivos, a primeira, foi adicionar NodePort no arquivo do serviço do Jenkins, com a finalidade de definir uma porta fixa para o serviço.

Ao subir esse pod, os dados não ficavam persistentes e para corrigir isso, foi feito um ssh para o minikube e foi criado uma pasta denominada Jenkins\_home no diretório /data. O comando utilizado foi mkdir /data/Jenkins\_home. Com essa pasta criada, modificados o arquivo .yaml referente ao deployment do Jenkins, foi adicionado um hostPath associado ao volume, a path foi a pasta criada no minikube.

Com essas duas alterações, o serviço ficou com uma porta fixa e os dados ficaram persistentes.

4. **Todos os scripts e manifests do Wordpress:**

<https://github.com/GabrielFerreira7/compassFinal>

- No anexo a seguir apresenta os PODs ativos, incluindo o do Jenkins.

```
(gabriel@localhost trabalho)$ kubectl get po -A
```

NAMESPACE	NAME	READY	STATUS	RESTARTS	AGE
kube-system	coredns-64897985d-mcngg	1/1	Running	0	154m
kube-system	etcd-minikube	1/1	Running	2	154m
kube-system	kube-apiserver-minikube	1/1	Running	2	154m
kube-system	kube-controller-manager-minikube	1/1	Running	2	154m
kube-system	kube-proxy-69rd2	1/1	Running	0	154m
kube-system	kube-scheduler-minikube	1/1	Running	2	154m
kube-system	storage-provisioner	1/1	Running	5 (4m35s ago)	154m
proj-compass	jenkins-789c9b6b84-6tc1f	1/1	Running	0	146m

```
(gabriel@localhost trabalho)$ _
```

- Neste próximo anexo, é mostrado os IPs das máquinas:

```
TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

docker0: flags=4099<UP,BROADCAST,MULTICAST> mtu 1500
    inet 172.17.0.1 netmask 255.255.0.0 broadcast 172.17.255.255
    ether 02:42:ce:51:03:38 txqueuelen 0 (Ethernet)
    RX packets 0 bytes 0 (0.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 0 bytes 0 (0.0 B)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

veth3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.0.130 netmask 255.255.255.0 broadcast 192.168.0.255
    inet6 fe80::a00:27ff:fee3:5ee2 prefixlen 64 scopeid 0x20<link>
    ether 08:00:27:e3:5e:e2 txqueuelen 1000 (Ethernet)
    RX packets 321820 bytes 384013054 (366.2 MiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 220572 bytes 145720972 (138.9 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Loopback Local)
    RX packets 2453 bytes 350872 (342.6 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 2453 bytes 350872 (342.6 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

veth84f5fc7: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet6 fe80::203d:a6ff:fed7:f704 prefixlen 64 scopeid 0x20<link>
    ether 22:3d:a6:d7:f7:04 txqueuelen 0 (Ethernet)
    RX packets 178204 bytes 143815177 (137.1 MiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 244305 bytes 379528330 (361.9 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

(gabriel@localhost trabalho)$ _
```

- Sobre o Jenkins:
  1. Exibindo o Jenkins no navegador do Windows:

Uma das maiores dificuldades, foi exibir o Jenkins no navegador, justamente, por conta de o Jenkins estar rodando dentro do minikube que por sua vez está dentro de uma VM.

Para resolver isso, foi utilizado o comando iptables. Com esse comando, foi possível, criar rotas de acesso.

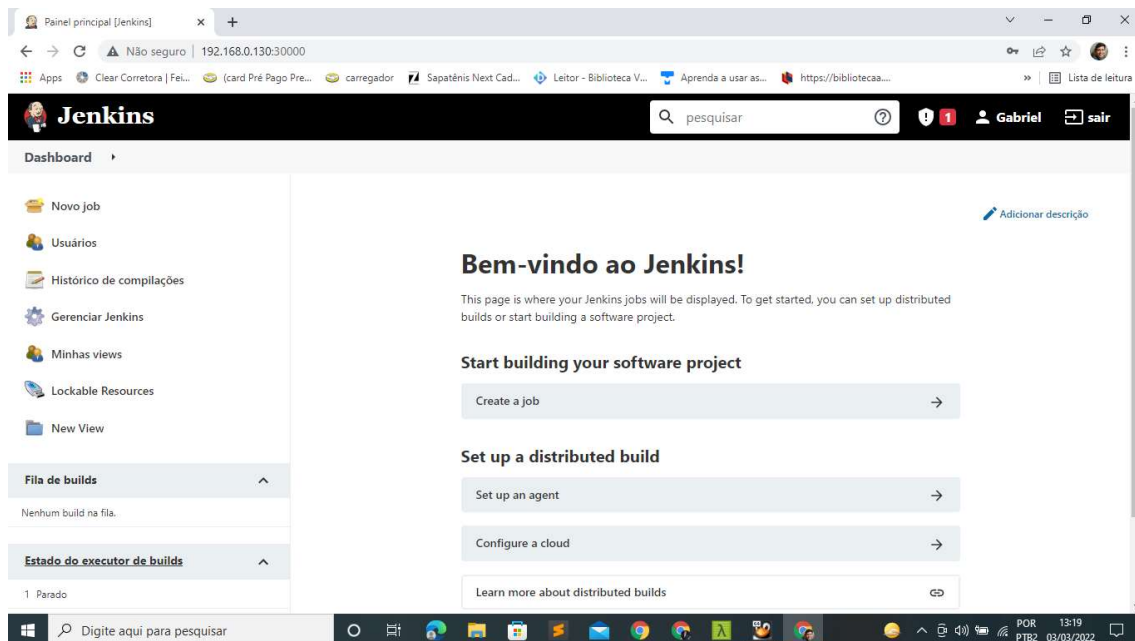
Primeiro foi feito um redirecionamento, da porta 3000 da VM para o minikube na porta 3000. O comando utilizado foi:

```
sudo iptables -A PREROUTING -t nat -i enp0s3 -p tcp --dport 30000 -j DNAT --to 192.168.49.2:30000
```

sendo o 192.168.49.2 o ip do minikube.

Depois de criar esse redirecionamento, foi dada permissão ao minikube na porta 3000 através do comando: `sudo iptables -A FORWARD -p tcp -d 192.168.49.2 --dport 30000 -j ACCEPT`.

Com isso, no ambiente externo do Windows, quando tentamos acessar o ip da vm na porta 30000, o Oracle Linux faz o redirecionamento para o minikube na porta 30000, e assim conseguimos acessar o Jenkins.



## 2. Pipeline Jenkins:

A pipeline tem o objetivo de fazer a criação do deployment do wordpress, e para isso, foi necessário fazer um ssh do Jenkins para o oracle Linux 8.5.

Para fazer o ssh, foi necessário estabelecer uma relação de confiança entre o pod do Jenkins para o oracle Linux 8.5.

Com o pod do Jenkins rodando, foi executado o comando

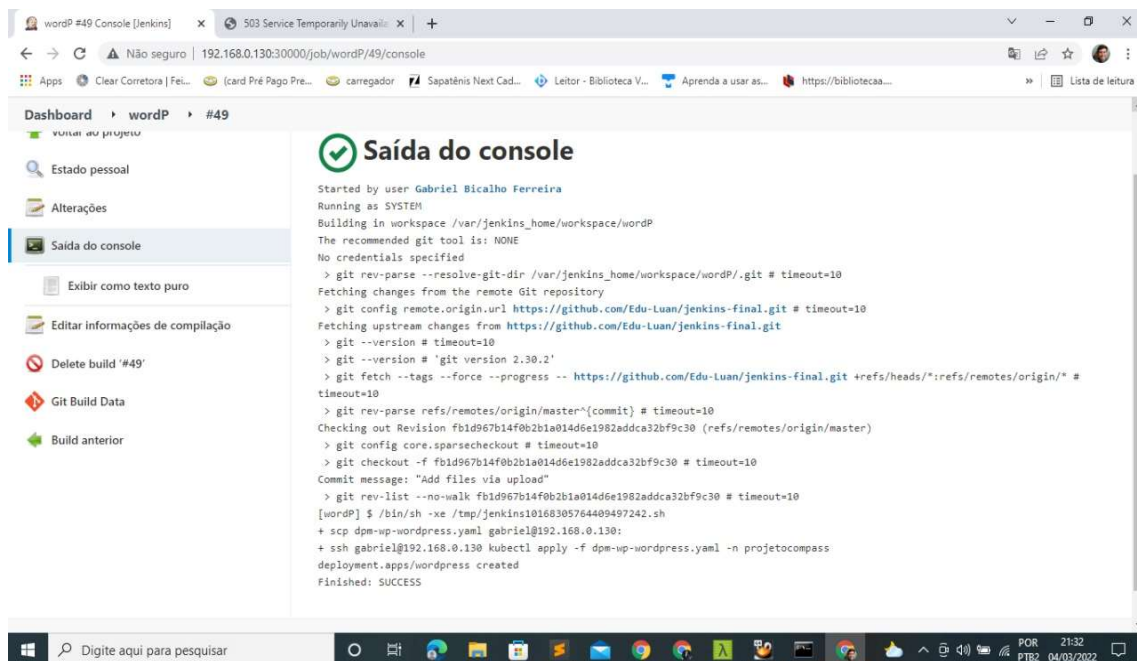
`Kubectl exec nome_pod -n namespace -it -- bash`, esse comando, permite entrar dentro do pod. Dentro do pod do Jenkins, foi criada uma chave na pasta `/var/Jenkins/home/.ssh` através do comando `ssh-keygen -t rsa`. Com esse comando foi criada duas chaves nesse diretório, uma pública e outra privada. Após isso, ainda dentro do pod foi executado o comando: `cat ~/.ssh/id_rsa.pub | ssh usuárioVM@ipVM 'cat - >> ~/.ssh/authorized_keys`. O que esse comando fez, foi enviar a chave pública para o usuário da vm, no diretório `~/.ssh` salvando em um arquivo denominado `authorized_keys`. Com isso, temos uma relação de confiança entre o pod do Jenkins e a vm, e agora o Jenkins pode conectar via ssh sem precisar de uma senha.

- **Sobre o Wordpress:**

1. **Manifest wordpress:**

O manifest wordpress, consiste em arquivos referente ao serviço, deployment, e volumes persistente do wordpress e do banco de dados mysql. Todos esses arquivos tiveram o deploy manual, exceto o deployment do wordpress, que será feito de forma automatizada usando o Jenkins.

2. **Log wordpress:**



### 3. Wordpress em funcionamento:

