Instalação e Configuração

Repositorio: https://github.com/rrs-rodrigues/Teste-CERTI/tree/main/1.Gestao%20de%20Servico%20Web

1 - Criação do projeto na Cloud

1.1 - Criar regras no firewall

- restringir o acesso público a instância apenas nas portas 80 e 443

1.2 - Criar vpn

- criar uma vpn para acesso do ip público da empresa no projeto onde a instância será criada

1.2 - Deploy da Instância

SO	Rocky Linux 8.10
RAM	4 GB
vCPU	4 vCPUs
Tipo de Armazenamento	SSD ou HD
Tamanho do Armazenamento	50 GB

2 - Particionamento

Ponto de montagem	Tamanho	Tipo	Sistema de Arquivos
swap	4 Gb	LVM	swap
/boot	800 Mb	Standard partition	xfs
/home	10 Gb	LVM	xfs
/	36 Gb	LVM	xfs

3 - Configuração

3.1 - Update

- Após instalação do SO, os pacotes e repositórios pré-instalados devem ser atualizados com o seguinte comando: **yum update**

3.2 - Instalação do Docker

Instalação do yum utils	yum install -y yum-utils
Adicionando o repositório do Docker	yum-config-manager --add-repo https://download.docker.com/linux/centos/docker-ce.repo
Instalação do docker, docker compose e suas dependências	yum install docker-ce docker-ce-cli containerd.io docker-compose-plugin -y
Reiniciando systemd para reconhecer os comandos do docker sem precisar reiniciar a instância	systemctl daemon-reload
Habilitando a inicialização do docker junto ao ao SO	systemctl enable docker
Iniciando o docker	systemctl start docker

3.3 - Configuração da Rede do Docker na Instância

- Essa parte é para evitar conflitos com a rede privada da empresa.
  3.3.1 - Criar um arquivo de nome **daemon.json** no diretório **/etc/docker/** com as seguintes linhas:

```
{
   "default-address-pools":
   [
    {"base":"10.10.0.0/16","size":24}
   ]
 }
```

  Obs: se houver conflito com a rede acima a mesma deve ser alterada.
  3.3.2 - Reiniciar o serviço do docker: **systemctl restart docker**

4 - Deploy da Aplicacao

4.1 - criação do **docker-compose.yml** em /opt/infra para isso crie o diretório com **mkdir /opt/infra/,** entre no diretório e crie o arquivo abaixo.

Obs: como yaml e orientado white space, oriento a baixa o arquivo docker compose do github que ele vai estar corretamente identado:
https://github.com/rrs-rodrigues/Teste-CERTI/blob/main/1.Gestao%20de%20Servico%20Web/docker-compose.yml

```
version: "3.8"

networks:
  infra:
   ipam:
    config:
     - subnet: 10.20.1.0/16

services:
  nginx:
   image: nginx:latest
   restart: always
   ports:
    - 80:80
    - 443:443
   networks:
    infra:
     ipv4_address: 10.20.1.2
   volumes:
    - /etc/localtime:/etc/locatime
    - ./default.conf:/etc/nginx/conf.d/default.conf


  novo-servico:
            image: #nome da imagem que sera usada
            restart: always
            networks:
              infra:
               ipv4_address: 10.20.1.3
            ports:
             - 3060 #porta que sera usada para comunicacao com nginx
            volumes:
              - /etc/localtime:/etc/localtime
               #insira aqui o volume da aplicacao que sera usada
```

Obs: iremos utilizar **restart: always** em todos os containers para que eles iniciem automaticamente se a instância for reiniciada e eles estiverem up. E estou passando **/etc/localtime** para sincronizar o horário do host com o container.

4.2 - Coloque o nome da imagem a ser usada em **image:** e adicione o volume que será usado no container do **novo-servico**.

4.3 - adicione no diretório o arquivo **default.conf** do nginx:

```
#novo-servico
upstream novo-servico{
        server 10.20.1.3:3060;
}

server {
        server_name #colocar aqui o endereço (seu dominio) cadastrado no DNS;
        listen 443;
        access_log /var/log/nginx/access.log;
        return 301 https://$host$request_uri;
        location / {
        proxy_pass http://novo-servico;
        }
}
```

4.4 - Subindo os container

Para subir os containers em background	docker compose up -d
Para ver o status dos containers do docker compose	docker compose ps
Para parar o docker compose	docker compose down

Obs: esses comandos só funcionam no diretório onde está o docker-compose.yml.

5 - Monitoramento

5.1 - adicionando repositório do zabbix

```
rpm -Uvh https://repo.zabbix.com/zabbix/6.0/rhel/8/x86_64/zabbix-release-6.0-5.el8.noarch.rpm
```

5.2 - Instalando zabbix-agent2

```
yum install zabbix-agent2 zabbix-agent2-plugin-*
```

5.3 - Configurando o zabbix-agent2: **/etc/zabbix/zabbix_agent2.conf**

```
PidFile=/run/zabbix/zabbix_agent2.pid
LogFile=/var/log/zabbix/zabbix_agent2.log
LogFileSize=0
Server=
ServerActive=
Hostname=
Include=/etc/zabbix/zabbix_agent2.d/*.conf
PluginSocket=/run/zabbix/agent.plugin.sock
ControlSocket=/run/zabbix/agent.sock
Include=./zabbix_agent2.d/plugins.d/*.conf
```

5.3.1 - adicione **Server** e **ServerActive** o IP ou URL do zabbix server. E em Hostname o nome do host como está cadastrado no zabbix server.

5.4 - Ativando zabbix agent2 e iniciando com o SO.

Habilitando inicialização com o SO	systemctl enable zabbix-agent2
Iniciando o serviço	systemctl start zabbix-agent2

5.5 - Adicionando o usuário do zabbix ao grupo do docker, isso é importante para o monitoramento dos containers

```
usermod -aG docker zabbix
```

5.5 - No zabbix server adicione os templates de monitoramento do docker e linux.