4525 – Curso de Infraestrutura Ágil com Práticas DevOps

Desafio - Laboratório Parte 1: Automação DNS

Neste desafio iremos utilizar os conhecimentos que tivemos no desenvolvimento do primeiro laboratório da Dexter com automação do serviço DNS. Nesse desafio vamos melhorar o teste feitos no serviço DNS antes de entrar em produção e realizar a configuração de um segundo DNS para a nossa Infraestrutura.

Parte 1. Melhorando teste automatizados do Serviço DNS.

Ao instalar os serviço do DNS (bind9), o mesmo traz alguns comandos para ajudar no desenvolvimento do arquivos de configurações realizando teste dos arquivos, esses comandos são **named-checkconf** utilizado para validar o arquivo de configuração, e o **named-checkzone** para validar o arquivo de zona do DNS, sendo assim o seu desafio será aumentar os teste automatizados que usamos na primeira parte do Laboratório. Utilize os manuais do linux ou pesquisa pela internet para utilizar os dois comandos descritos acima.

(Obs: É sempre bom realizar o teste tanto no ambiente de homologação, quando no ambiente de produção para sempre validar os dois ambientes e evitar problemas de automação.)

Para melhorar os teste do serviço dns podemos utilizar o comando named-checkconf para validar o arquivo de configuração e podemos utilizar o comando named-checkzone para validar a configuração da zone de dns:

No arquivo: /etc/ansible/playbooks/homolog/dns/docker.yml foi adcionado dois comando

para realizar os teste de configuração garantida pelo módulo do puppet, que estão em destaque abaixo:

```
---
# craete and test containers

- name: testing docker container
  command: docker inspect ns-homolog
  register: container
  ignore_errors: yes

- name: stop docker container
  command: docker stop ns-homolog
  when: container.rc == 0

- name: remove docker container
  command: docker rm ns-homolog
  when: container.rc == 0

- name: create docker container
  command: docker run -dit --name ns-homolog --hostname ns-homolog --add-host=puppet:192.168.200.50 homolog-template /bin/bash

- name: running puppet agent
  command: docker exec ns-homolog puppet agent -t --environment=homolog
  ignore_errors: yes

- name: testing configuration file
  command: docker exec ns-homolog named-checkconf /etc/bind/named.conf

- name: testing zone files
  command: docker exec ns-homolog named-checkzone dexter.com.br
```

/var/cache/bind/db.dexter

```
- name: testing docker container
  command: docker exec ns-homolog service bind9 status

- name: stop docker container
  command: docker stop ns-homolog

- name: remove docker container
  command: docker rm ns-homolog
```

No arquivo: /etc/ansible/playbooks/production/dns/docker.yml foi adcionado dois comando para realizar os teste de configuração garantida pelo módulo do puppet, que estão em destaque abaixo:

```
---
# Playbook: create docker container

- name: starting docker container
  command: docker start ns1
  ignore_errors: yes
  register: result


- name: creating docker container
  command: docker run -dit --name ns1 --hostname ns1 --add-host=puppet:10.8.0.6 --dns-search=dexter.com.br production-template /bin/bash
  when: result.rc != 0

- name: running puppet agent
  command: docker exec ns1 puppet agent -t --environment=production
  ignore_errors: yes
```

- name: testing configuration files
command: docker exec ns1 named-checkconf /etc/bind/named.conf

- name: testing configuration file
command: docker exec ns1 named-checkzone dexter.com.br /var/cache/bind/db.dexter

- name: testing dns service
command: docker exec ns1 service bind9 status

Parte 2. Configurando uma segunda instância do DNS.

Por questão de alta disponibilizada e muito comum configurar mais de uma servidor DNS dentro do ambiente, e caso um dos servidores não responder a própria máquina cliente tenta resolver pelo outros servidores. Então o seu desafio e provisionar mais um container com serviço DNS. Sendo que o processo de homologação e produção deve sempre configurar dois containers do Docker, ou seja o mesmo playbook criar sempre dois DNS.

Dica: É preciso configurar o node dentro do puppet para pegar as configurações quando o agent do puppet for executado dentro do ambiente de produção e do ambiente de homologação.

Dica: Para deixar mais simples o Playbook, você pode criar um arquivo chamado docker02.yml, e simplesmente realizar o include desse arquivo, sendo assim o playbook fica mais limpo e mais fácil para realizar a manutenção do Código.

Dica: Como nós realizamos o deploy automático do container e as configurações que são garantindas com puppet, nesse caso nem é preciso configurar um servidor do tipo slave. Porque atráves do puppet sempre garantimos o mesmo arquivos para os containers. Porém em um ambiente de produção sempre e recomendado que os outros servidores de DNS sejam slaves do servidor DNS Master da Infraestrutura.

Parte 2. Configurando uma segunda instância do DNS.

Precisamos especificar ao puppet a configuração do novo node para o ambiente de produção que é um container na AWS:

No arquivos: /etc/puppet/environments/manifests/site.pp

```
#node 'ns1.dexter.com.br' {
#   include puppet-dns
#}


# Regex para selecionar ns1, ns2, ns3, nsN
node /^ns\d+.dexter.com.br$/ {
   include puppet-dns
}
```

OBS: Foi utilizado uma expressão regular para selecionar o modulo puppet-dns para qualquer node que possua o nome nsN, sendo que N é um numero então ns1, ns2, ns3 e assim por diante.

Foi adicionar um novo arquivo chamado docker02.yml para provisionar o segundo container com o DNS da Dexter.

No arquivo: /etc/ansible/playbooks/production/dns/main.yml foi realizado a mudança que está em destaque a baixo

```
---
# Playbook: Create docker container production

- name: docker create container
  hosts: production
  remote_user: ubuntu
  sudo: yes

  tasks:
    - include: docker.yml
     - include: docker02.yml
```

Foi adicionar um novo arquivo chamado docker02.yml para provisionar o segundo container com o DNS da Dexter.

No arquivo: /etc/ansible/playbooks/production/dns/docker02.yml:

```
---
# Playbook: create docker container

- name: starting docker container
  command: docker start ns2
  ignore_errors: yes
  register: result

- name: creating docker container
  command: docker run -dit --name ns2 --hostname ns2 --add-host=puppet:10.8.0.6 --dns-search=dexter.com.br production-template /bin/bash
  when: result.rc != 0

- name: running puppet agent
  command: docker exec ns2 puppet agent -t --environment=production
```

```
  ignore_errors: yes

- name: testing configuration files
  command: docker exec ns2 named-checkconf /etc/bind/named.conf

- name: testing configuration file
  command: docker exec ns2 named-checkzone dexter.com.br /var/cache/bind/db.dexter

- name: testing dns service
  command: docker exec ns2 service bind9 status
```