Redes e Sistemas Distribuídos

Simulando sistemas distribuídos com Docker

Você sabia que seu material didático é interativo e multimídia? Isso significa que você pode interagir com o conteúdo de diversas formas, a qualquer hora e lugar. Na versão impressa, porém, alguns conteúdos interativos ficam desabilitados. Por essa razão, fique atento: sempre que possível, opte pela versão digital. Bons estudos!

Nesta webaula, apresentaremos os comandos Docker que podem ser utilizados em sistemas distribuídos, por exemplo, comandos para contêineres, cluster e de monitoramento. Vale salientar que a leitura do livro didático é indispensável, uma vez que os assuntos são tratados de forma aprofundada.

Docker

É uma famosa plataforma genérica de conteinerização, conceito parecido com virtualização, porém considerado "mais leve" que este. Atualmente, contêineres são populares devido à facilidade e à flexibilidade que advêm de seu uso. Assim, chegou o momento de aprendermos a utilizar essa famosa ferramenta.

Instalação do Docker

Todo o procedimento de instalação deve ser feito com um usuário com permissões de "administrador". No caso que será apresentado a seguir, utilizaremos o sistema operacional Ubuntu (versão mais atualizada 14.04.5 LTS) e o "root", por meio do comando "sudo su". Em resumo, o processo de instalação é bem simples, sendo somente necessário avançar as seguintes etapas:

Etapa 1

Antes de instalar a ferramenta, devemos remover versões anteriores do Docker que possam estar instaladas, utilizando o seguinte comando:

```
sudo apt-get remove docker docker-engine docker.io
```

Caso não tenha nenhuma versão instalada, será exibida a mensagem que foi impossível encontrar o pacote docker-engine.

Etapa 2

Antes de instalar o Docker CE pela primeira vez em uma nova máquina host, você precisa configurar o repositório do Docker, atualizando os pacotes de sua máquina. Para isso, são necessários o seguinte comando:

```
sudo apt-get update
```

Este comando tem o objetivo de realizar uma atualização de pacotes do Ubuntu.

Etapa 3

Através do seguinte comando atualizamos os pacotes necessários para a instalação do Docker.

```
sudo apt-get install \ apt-transport-https \ ca-certificates \ curl \ software-properties-common
```

Etapa 4

Após atualizarmos os repositórios, adicionaremos o repositório de instalação do Docker. Para isso, é necessário o uso do seguinte comando:

```
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -
```

Com esse comando apontamos o caminho de instalação oficial do Docker, no qual o Ubuntu deve acessar quando fizermos a instalação.

Etapa 5

Após o apontamento da URL onde está disponível para download o Docker para Ubuntu, será possível adicionar o repositório no próximo comando:

```
sudo add-apt-repository \ "deb [arch=amd64] https://download.docker.com/linux/ubuntu
\ $(lsb_release -cs) \  stable“
```

O comando utiliza a permissão considerada “admin” nos sistemas operacionais da família Linux, através da palavra “sudo”.

Etapa 6

Depois de utilizar a permissão de usuário do sudo, utilizamos o “add-apt-repository” para adicionar o repositório, que pode ser comparado a uma loja de aplicativos, responsável pelo download do Docker na versão Ubuntu. O restante do comando é o caminho do repositório.

Etapa 7

Após adicionar o repositório para download do Docker, devemos mais uma vez atualizar o “apt-get”, conforme o comando seguinte para aplicar as alterações:

```
sudo apt-get update
```

Etapa 8

Neste momento, utilizaremos o comando “sudo” de instalação do Docker, para utilizar a permissão “admin” do sistema e o “apt-get install” para fazer a instalação. Por fim, deve-se inserir o nome do programa que será instalado, no caso, o Docker (“docker-ce”).

```
sudo apt-get install docker-ce
```

Com todas configurações feitas para atualizar os pacotes necessários e adicionar o repositório que contém o Docker, agora é possível fazer a instalação.

Iniciando e testando o Docker

Para ver se o Docker foi instalado corretamente, devemos iniciar o serviço do Docker e, após isso, verificar se ele está em execução. Para tanto, é necessário utilizar os comandos: *sudo*, *service docker start* e *service docker status*

Agora que instalamos e verificamos seu funcionamento, o sistema está apto a receber as especificidades que queremos criar. Para isso, é possível usar o Docker Swarm. Nesse cenário é possível agrupar vários hosts em um mesmo pool de recursos, o que facilita o deploy de containers (DIEDRICH, 2018).

Após a criação do nosso login na plataforma, teremos acesso a uma interface para criação de instâncias como clusters e nós.

Por meio dos comandos que serão apresentados a seguir, podemos simular algumas características de sistemas distribuídos com Docker.

Comando executado “fora” do(s) nó(s)

Um dos comandos importantes é o de **criação de um novo manager para o nosso cluster**. Através dele, criamos um manager chamado de “mestre”:

```
docker-machine create --driver virtualbox mestre
```

Ao executarmos esse comando, o começo dele significa que estamos criando uma nova máquina através do Docker (“docker-machine create”).

Comando executado “dentro” do(s) nó(s) manager

O comando a seguir pode ser utilizado para iniciar o cluster através do framework swarm, de gerenciamento de contêineres.

```
docker swarm init --advertise-addr < IP DO MESTRE >
```

O comando a seguir pode ser utilizado para iniciar o cluster através do framework swarm, de gerenciamento de contêineres.

“node ls”

Podemos consultar os nós que fazem parte do cluster utilizando o comando “node ls”:

```
docker node ls
```

“inspect”

Em algumas ocasiões, precisamos verificar informações sobre um nó específico. Para isso, utilizamos o comando Docker “inspect”. No exemplo a seguir trazemos informações sobre o nó chamado “escravo1”:

```
docker inspect escravo1
```

“ps”

Depois de criar o serviço de internet chamado “WEB”, podemos utilizar o comando “ps”, seguido do nome do serviço, para verificar suas informações. Por exemplo:

```
docker service ps WEB
```

“update”

Podemos utilizar o comando “update” para alterar a versão do “Nginx”. Por exemplo, é possível alterar a versão 1.12.1 para a 1.13.5:

```
docker service update --image nginx:1.13.5 WEB
```

Comando executado “dentro” do(s) nó(s) worker

O comando a seguir pode ser utilizado para adicionar um nó escravo (worker) ao nosso cluster. Para isso, devemos acessar o nó escravo que queremos adicionar a um cluster e executar o comando a seguir, passando o token de segurança e o IP do nó mestre (manager), seguido por porta, conforme a sintaxe representada no comando:

```
docker swarm join --token < IP do mestre:Porta >
```

Comando executado “dentro” de cada UM dos nós (managers e workers)

Quando executamos o comando Docker “system prune” com o parâmetro “all” dentro de um nó, ele será responsável por apagar/deletar tudo o que foi feito dentro do mesmo. Conforme observamos a seguir sua utilização:

```
docker system prune –all
```

Para terminar os estudos desta webaula, vamos aprender a orquestrar o servidor web Apache em um cluster simples. Primeiramente, você deve estar logado na plataforma de “playground” do Docker. Em seguida, será necessário:

Etapa 1

Criar um cluster com 3 nós, que serão suficientes para analisarmos nosso cluster sem comprometer a usabilidade da plataforma de testes do Docker. Sendo assim, você deve adicionar 3 nós através do botão “Add new instance”.

Etapa 2

No nó que você deseja que seja o mestre, digite o seguinte comando:

docker swarm init --advertise-addr < endereço IP desse nó >

Ao executar esse comando, é apresentada uma saída, com a mensagem: “Para adicionar um worker ao swarm, execute o seguinte comando”, a qual você deve copiar. Esse comando deverá ser executado em cada um dos demais nós do cluster, adicionando cada um deles como workers desse cluster.

Etapa 3

Agora que os nós estão criados e seus papéis definidos, para criar o serviço que estará rodando (de maneira distribuída, replicada) do servidor web Apache, digite o seguinte comando no nó mestre:

docker service create --name WEB --publish 80:80 --replicas=5 httpd

Etapa 4

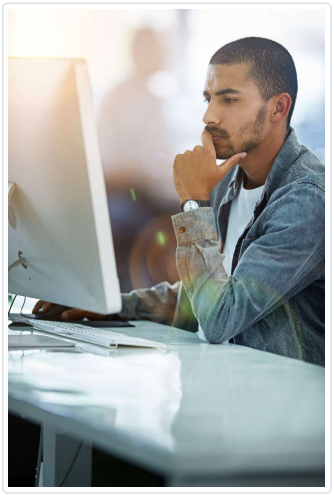
Para saber em quais nós as 5 réplicas desse serviço estão sendo executadas, digite o seguinte comando:

docker service ps WEB

Etapa 5

Por fim, precisamos acessar a página de boas-vindas do servidor Apache através do(s) endereço(s) IPv4 de cada nó onde esse serviço web estiver rodando. Para acessar esta página, basta clicar na porta que foi mapeada por você na parte superior (que aparece como um hyperlink), em cada um dos nós onde esse serviço está rodando (no meu caso, nós 1 e 2 do cluster), para vermos a famosa mensagem “It works!” do Apache.

Esta sequência de comandos que utilizamos para orquestrar um servidor web Apache em 3 nós é a configuração utilizada em sistemas distribuídos para que os acessos há um website sejam balanceados. Vale ressaltar que, caso ocorra algum problema em um dos nós que mantém a aplicação, o outro nó assume a execução.



Concluindo esta webaula, esperamos que você tenha tido um bom proveito dos conteúdos apresentados!

Bons estudos!

Para visualizar o vídeo, acesse seu material digital.