04

Instalando dependências

Transcrição

Criamos o nosso primeiro playbook, realizamos a sua execução, vimos como o arquivo fica dentro da máquina virtual, como é compartilhado do lado de fora. Agora vamos efetivamente instalar o projeto.

Para isto, colocaremos PHP, MySQL, Apache e Wordpress para funcionar. Veremos como o Ansible irá nos ajudar nessa tarefa.

Primeiramente, modificaremos o nosso playbook. Usamos o módulo `shell` para rodar comandos arbitrários na nossa máquina, agora o que queremos é modificar a instalação de dentro do Linux no caso, trata-se de uma instalação de Ubuntu.

O módulo que o Ansible usará é `apt`, mesmo nome do administrador de pacote das distribuição do tipo debian. Nós teremos o módulo `apt` para administrar pacotes do Ubuntu, do Debian, teremos o módulo `yum` para administrar pacote do `Red Hat` e do `CentOS`.

Teremos ainda mais uma coleção de módulos administradores de pacote. Neste momento, focaremos no `apt`. Com ele, nós podemos passar um parâmetro chamado `name`, referente ao pacote que desejamos instalar. Começaremos pelo `php5`.

```
- hosts: all
  tasks:
    - apt:
        name: php5
        state: latest
      become: yes
```

Na maior parte dos módulos do Ansible, é relevante usar o parâmetro `state` para informar o estado desejado da execução de uma task. No Ansible, toda task é checada para verificarmos qual foi o estado alcançado depois da execução.

Dependendo do módulo, teremos algumas opções de estado. Neste ponto, definiremos como `latest`, com isso garantimos que o Ansible vai definir o pacote mais recente de PHP5 na `apt`. Ou seja, se ele encontrar uma versão anterior, ela será atualizada.

E se nós estamos instalando um pacote em um sistema operacional, deve ser feito como um usuário root. No Ansible, informamos que queremos executar algo como root e não como o usuário que logamos via SSH, como o parâmetro `became`. Iremos configurá-lo com `yes` e deixá-lo no mesmo nível de `tasks` - ou seja, ele não é outro parâmetro dentro dele.

O que isso significa quando rodamos isso manualmente dentro do Ubuntu?

Em seguida, no Terminal, rodaremos a nossa máquina virtual para entendermos qual é o comando análogo.

```
vagrant@vagrant-ubuntu-trusty-64:~$ sudo apt-get install php5 -y^C
```

O mesmo código que escrevemos da task `apt` para instalarmos o PHP5, é equivalente para fazermos um `apt-get install` para o PHP5, como `sudo` , porque teremos que rodar com roots. Desta forma, Linux nos deixará alterar o filesystem e instalar um pacote. No entanto, não rodaremos o comando desta forma, faremos o `logout` e rodaremos o comando como Ansible, que é como ele funciona.

```
vagrant@vagrant-ubuntu-trusty-64:~$ sudo apt-get install php5 -y^C
vagrant@vagrant-ubuntu-trusty-64:~$ logout
Connection to 127.0.0.1 closed.
$ ansible-playbook provising.yml -u vagrant -i hosts --private-key .vagrant/machines/wordpress/
```

Executaremos o playbook `provisiong.yml` novamente, informando o usuário `vagrant` e o nosso arquivo de inventário e a nossa chave privada de autenticação. Lembre-se de verificar se o arquivo foi salvo corretamente.

Após o comando ser executado, veremos que a saída é diferente, notamos isso quando executamos a task `apt` , ele nos informa que ela é executada. Por ter que fazer download, verificar se `apt` está atualizada, a task será um mais demorada.

No fim do processo, teremos o seguinte retorno:

```
TASK [apt] *********************************************************************************
changed: [172.17.177.40]
```

O Ansible marcou `changed` para a task `apt` . Podemos confirmar se obtivemos sucesso é rodá-la novamente.

```
TASK [apt] *********************************************************************************
ok: [172.17.177.40]
```

Observe que dessa vez, a task será marcada com `ok` . Isto aconteceu porque o pacote já está instalado e está na versão que o Ansible precisa e, por isso, ele não executará de novo. Vale ressaltar que o Ansible utiliza o **princípio da Idempotência**, com qual apenas irá alterar o resultado quando as operações forem efetivamente alteradas.

Se ele garantiu que algo está no estado definido por você, ele não vai realizar a mesma operação uma segunda vez. A vantagem disso é que com isso, **evitamos** sobrescrever informações importante. Também evitamos chegar em uma estado indeterminado. O Ansible trabalha de duas formas:

- Ele garante que o processo será realizado da maneira como você determinou;
- Ou ele para de funcionar e não tenta executar tarefas em um estado não solicitado.

Nós conseguimos instalar o PHP5, o próximo passo será instalar o Apache. Antes, porém, realizaremos uma ação importante, a qual deveria ser repetida sempre que criamos uma task nova no playbook. Nós adicionaremos uma descrição, pensando no estágio em que teremos um número imenso de tasks no playbook. Como saberemos qual a utilidade de cada uma delas? Para evitarmos a confusão, escreveremos a ação solicitada no comando. No caso, a task criada será responsável por instalar o PHP5.

```
- hosts: all
  tasks:
    - name: 'Instala o PHP5'
```

```
      apt:
        name: php5
        state: latest
      become: yes
```

A seguir, usaremos o mesmo princípio para instalar o Apache2. Começaremos adicionando `name` e usaremos o mesmo módulo. O nome do pacote será `apache2` .

```
- hosts: all
  tasks:
    - name: 'Instala o PHP5'
      apt:
        name: php5
        state: latest
      become: yes
    - name: 'Instala o Apache2'
      apt:
        name: apache2
        state:lastest
      became: yes
```

Novamente, teremos que fazer isso como root, por isso, configuraremos `become` com `yes` . Em seguida, executaremos o Ansible.

```
$ ansible-playbook provising.yml -u vagrant -i hosts --private-key .vagrant/machines/wordpress/
```

De acordo com o retorno, o pacote do Apache2 já estava instalado.

```
TASK [Instala o Apache2] *******************************************************************
ok: [172.17.177.40]
```

Por que tivemos essa saída? No Ubuntu, o PHP5 depende do Apache2. Assim que instalamos o PHP5, automaticamente, será instalado o módulo de compatibilidade do Apache2. O PHP é usado, normalmente, para desenvolvimento web. O formato dos pacotes na APT, entende que se instalamos um queremos a instalação do outro.

Isto significa que não precisamos instalar o pacote do Apache2? Pensar assim não é uma boa prática. Imagine que isso seja modificado no futuro e o servidor padrão para trabalharmos com PHP passe a ser outro, por exemplo, o PHP-FPM - que vai melhorando sua reputação dentro do PHP.

Se você não se preocupa no instalação no módulo, correrá o risco de que seu playbook deixe de funcionar. É importante garantir a instalação de todas as dependências que são necessárias para o seu ambiente funcionar, estando ou não presentes. Para isto, elas devem ser citadas nominalmente,deixando explícito o que dever ser feito.

Depois, o MOD PHP será o próximo que instalaremos, ele será o módulo de integração entre o PHP e o Apache.

```
- hosts: all
  tasks:
    - name: 'Instala o PHP5'
```

```
    apt:
      name: php5
      state: latest
    become: yes
  - name: 'Instala o Apache2'
    apt:
      name: apache2
      state:lastest
    became: yes
  - name: 'Instala o Apache2'
    apt:
      name: libapache2-mod-php5
      state:lastest
    became: yes
```

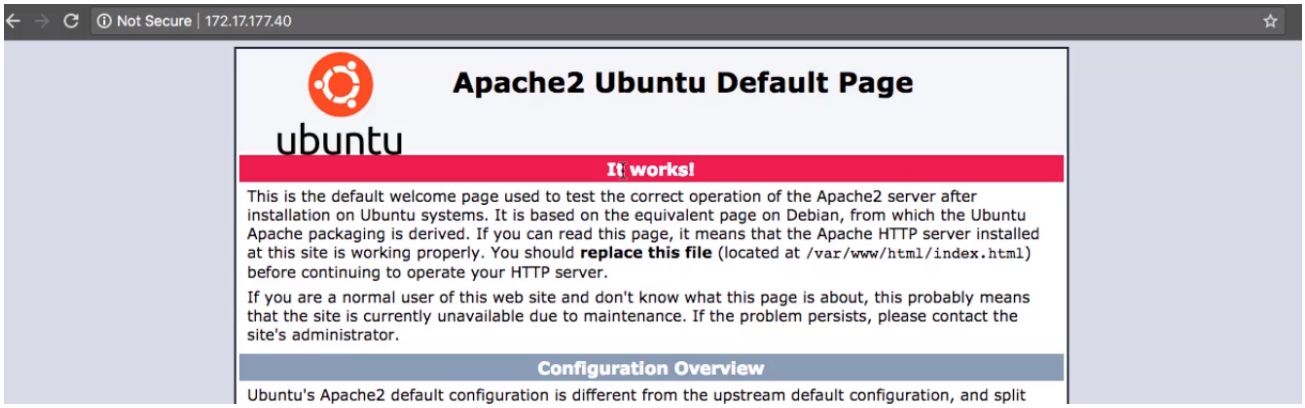
Independente se ele já estiver instalado - porque instalamos o Apache e o PHP -, devemos garantir que os módulos utilizados estão disponíveis, no estado que são necessários.

Em seguida, verificaremos se funcionou:

```
$ ansible-playbook provising.yml -u vagrant -i hosts --private-key .vagrant/machines/wordpress/

PLAY [all] *********************************************************************

TASK [Gathering Facts] *********************************************************
ok: [172.17.177.40]

TASK [Instala o PHP5] **********************************************************
ok: [172.17.177.40]

TASK [Instala o Apache2] *******************************************************
ok: [172.17.177.40]

TASK [Instala o modphp] ********************************************************
changed: [172.17.177.40]

PLAY RECAP *********************************************************************
172.17.177.40                  : ok=4    changed=1    unreachable=0    failed=0
```

Desta vez, o módulo teve que ser instalado. Após a finalização do processo, e após rodarmos o Ansible novamente, garantiremos que ele não irá reinstalar o MOD PHP. Até aqui, nosso ambiente tem PHP5, Apache2 e o MOD PHP (a integração entre os dois está funcionando).

Agora, para conferirmos se a nossa instalação de Apache funciona, acessaremos pelo browser o ip da máquina virtual (`172.17.177.40`). Veremos uma tela com o logo do Ubuntu.



Vemos que o Apache está funcionando.

Mais adiante, veremos como melhorar o playbook, que ainda não tem algumas dependências do PHP e se evitaremos deixar esse processo de instalação repetitivo. Seria trabalho e desnecessário mostrarmos no curso a criação dessas linhas de código. Nosso usaremos um loop chamado `with_items` , que usaremos para construir de forma mais inteligente essa parte de administração de dependência.