

Trabalhando com vários logins

Transcrição

Anteriormente, descobrimos como era possível simplificar o playbook usando o loop `with_items` e incluímos as dependências que faltavam para o ambiente funcionar. Agora veremos como é possível configurar o arquivo de inventário para suprir usuários diferentes, chaves privadas diferentes. No nosso caso, simplificaremos a forma como estamos rodando o comando Ansible playbook.

Incluimos as seguintes dependências:

- `apache2`
- `libapache2-mod-php5`
- `php5-gd`
- `libssh2-php`
- `php5-mcrypt`
- `mysql-server-5.6`
- `python-mysqldb`
- `php5-mysql`

Estas já devem estar instaladas na nossa máquina para seguirmos com o trabalho. Em seguida, no arquivo de `hosts`, incluiremos duas informações: **nome do usuário**, que estamos passando na linha de comando, e a **chave privada**. Sempre que executamos um comando para o Ansible, nós passamos essas duas informações.

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

O primeiro é sempre o usuário `ssh` (que é o método de conexão do Ansible), enquanto o segundo é o método pelo qual o usuário se loga e foi criado automaticamente pelo Vagrant. Para diminuirmos o tamanho do comando, incluiremos duas variáveis na mesma linha onde está localizado o IP do servidor. Elas serão adicionadas na seguinte ordem:

- Nome do usuário

Em `hosts`, incluiremos o nome do usuário no código.

```
[wordpress]  
172.17.177.40 ansible_user=vagrant
```

Desta forma, será **desnecessário** passar o usuário na linha de comando.

```
$ ansible-playbook provisioning.yml -i hosts --private-key .vagrant/machines/wordpress/virtualb
```

Observe que removemos `-u vagrant` do comando executado anteriormente. Se realizarmos a execução, ele continuará funcionando como antes.

- Chave privada

Também vamos adicionar a informação da chave privada.

É um caminho bastante extenso e tem caracteres entre os termos, é útil colocá-los entre aspas e assim garantir que a string não será quebrada. No arquivo `hosts` escreveremos:

```
[wordpress]
172.17.177.40 ansible_user=vagrant ansible_ssh_private_key_file="/Users/<seu_usuario>/wordpress_
```

Nesta etapa, você deverá ter **atenção**, porque o seu caminho será diferente do meu. Será preciso pesquisar a chave na sua máquina.

No terminal, digitaremos o seguinte:

```
$ pwd
/Users/<seu_usuario>/wordpress_com_ansible
$ ls .vagrant/machines/wordpress/vistualbox/private_key
```

Agora, se executarmos o comando, teremos que passar apenas o playbook e o inventário - informações que sempre serão solicitadas:

```
$ ansible-playbook provisioning.yml -i hosts
```

O comando ficou **mais simples**. Existe uma grande vantagem para essa abordagem em que colocamos o nome de usuário, chave privada, além de outras configurações possíveis do Ansible. Imagine se precisássemos configurar uma segunda máquina e o usuário tivesse um **nome diferente** de Vagrant, ou seja, não é uma máquina virtual. Como conseguiríamos fazer funcionar na linha de comando:

```
[wordpress]
172.17.177.40 ansible_user=vagrant ansible_ssh_private_key_file="/Users/<seu_usuario>/wordpress_
172.17.177.42 ansible_user=admin
```

Só funcionaria se passássemos o nome de usuário correto. Mas passando por dentro, conseguimos ter cada máquina com a configuração necessária. Desta forma, que garantimos o bom funcionamento de um ambiente heterogêneo no Ansible.

Está faltando apenas o banco de dados. A seguir, veremos como configurar o MySQL e preparar a nossa instalação de Wordpress.