Vagrant Guide

Introduction

Vagrant is a tool to manage virtual machine environments, and allows you to configure and use reproducible work environments on top of various virtualization and cloud platforms. It also has integration with Ansible as a provisioner for these virtual machines, and the two tools work together well.

This guide will describe how to use Vagrant 1.7+ and Ansible together.

If you're not familiar with Vagrant, you should visit the documentation.

This guide assumes that you already have Ansible installed and working. Running from a Git checkout is fine. Follow the :ref:`installation_guide` guide for more information.

> **System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\ansible-devel\docs\docsite\rst\scenario_guides\[ansible-devel][docs][docsite][rst][scenario_guides]guide_vagrant.rst`, line 20); *backlink***
>
> Unknown interpreted text role "ref".

Vagrant Setup

The first step once you've installed Vagrant is to create a `Vagrantfile` and customize it to suit your needs. This is covered in detail in the Vagrant documentation, but here is a quick example that includes a section to use the Ansible provisioner to manage a single machine:

```
# This guide is optimized for Vagrant 1.8 and above.
# Older versions of Vagrant put less info in the inventory they generate.
Vagrant.require_version ">= 1.8.0"

Vagrant.configure(2) do |config|

  config.vm.box = "ubuntu/bionic64"

  config.vm.provision "ansible" do |ansible|
    ansible.verbose = "v"
    ansible.playbook = "playbook.yml"
  end
end
```

Notice the `config.vm.provision` section that refers to an Ansible playbook called `playbook.yml` in the same directory as the `Vagrantfile`. Vagrant runs the provisioner once the virtual machine has booted and is ready for SSH access.

There are a lot of Ansible options you can configure in your `Vagrantfile`. Visit the Ansible Provisioner documentation for more information.

```
$ vagrant up
```

This will start the VM, and run the provisioning playbook (on the first VM startup).

To re-run a playbook on an existing VM, just run:

```
$ vagrant provision
```

This will re-run the playbook against the existing VM.

Note that having the `ansible.verbose` option enabled will instruct Vagrant to show the full `ansible-playbook` command used behind the scene, as illustrated by this example:

```
$ PYTHONUNBUFFERED=1 ANSIBLE_FORCE_COLOR=true ANSIBLE_HOST_KEY_CHECKING=false ANSIBLE_SSH_ARGS='-o UserKnownHostsFile=/dev/null -o Ident
```

This information can be quite useful to debug integration issues and can also be used to manually execute Ansible from a shell, as explained in the next section.

Running Ansible Manually

Sometimes you may want to run Ansible manually against the machines. This is faster than kicking `vagrant provision` and pretty easy to do.

With our `Vagrantfile` example, Vagrant automatically creates an Ansible inventory file in `.vagrant/provisioners/ansible/inventory/vagrant_ansible_inventory`. This inventory is configured according to the SSH tunnel that Vagrant automatically creates. A typical automatically-created inventory file for a single machine environment may look something like this:

> **System Message: WARNING/2 (`D:\onboarding-resources\sample-onboarding-resources\ansible-devel\docs\docsite\rst\scenario_guides\[ansible-devel][docs][docsite][rst][scenario_guides]guide_vagrant.rst`, line 102)**
>
> Cannot analyze code. No Pygments lexer found for "none".
>
> ```
> .. code-block:: none
>
>     # Generated by Vagrant
>
>     default ansible_host=127.0.0.1 ansible_port=2222 ansible_user='vagrant' ansible_ssh_private_key_file='/home/someone/coding-in-a
> ```

If you want to run Ansible manually, you will want to make sure to pass `ansible` or `ansible-playbook` commands the correct arguments, at least for the *inventory*.

```
$ ansible-playbook -i .vagrant/provisioners/ansible/inventory/vagrant_ansible_inventory playbook.yml
```

Advanced Usages

The "Tips and Tricks" chapter of the Ansible Provisioner documentation provides detailed information about more advanced Ansible features like:

- how to execute a playbook in parallel within a multi-machine environment
- how to integrate a local `ansible.cfg` configuration file

> **System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\ansible-devel\docs\docsite\rst\scenario_guides\[ansible-devel][docs][docsite][rst][scenario_guides]guide_vagrant.rst`, line 125)**
>
> Unknown directive type "seealso".
>
> ```
> .. seealso::
>
>     `Vagrant Home <https://www.vagrantup.com/>`_
>         The Vagrant homepage with downloads
>     `Vagrant Documentation <https://www.vagrantup.com/docs/>`_
>         Vagrant Documentation
>     `Ansible Provisioner <https://www.vagrantup.com/docs/provisioning/ansible.html>`_
>         The Vagrant documentation for the Ansible provisioner
>     `Vagrant Issue Tracker <https://github.com/hashicorp/vagrant/issues?q=is%3Aopen+is%3Aissue+label%3Aprovisioners%2Fansible>`_
> ```

```
    The open issues for the Ansible provisioner in the Vagrant project
:ref:`working_with_playbooks`
    An introduction to playbooks
```