Deploying Wordpress on an OpenStack VM by Executing a Single Command

Brent Gawryluik
University of British Columbia
Advanced Research Computing
About Me

- Compute Canada/Westgrid HPC systems administrator since 2003
- Designed, installed, & maintained the Glacier and Orcinus clusters
- Bare-metal computing, physical data centres, and hardware systems administration (98% of career)
- Learning to both accept virtualization and embrace the universe of the cloud
Intended Audience

- Researchers and students who:
  - Have Compute Canada user credentials
  - Have access to a Compute Canada OpenStack cloud installations
    - West Cloud or East Cloud
  - Are interested in running web-based services in the cloud
- If this is NOT you, no problem
  - apply your own changes and transform the code to suit your own cloud computing environment
Public Git Repository Location

https://github.com/brentg71/openstack-wordpress
Brief Project Introduction

“The entire purpose of this repository is to document, store, and maintain all of the "source code" that was necessary for me to deploy and provision a reasonably secure WordPress website on an OpenStack VM -- running either a CentOS or Ubuntu Linux distribution of my choosing -- by executing a single command.”

-- Me. Spring. 2016.
Project Assumptions

- You are comfortable using the Linux command line
- You have some experience with the following:
  - **Git** - to clone the source repository
  - **Vagrant** - to instantiate the VM
  - **OpenStack Cloud Provider** (Vagrant plugin) - to interface with OpenStack
  - **Ansible** - to provision and provide a tool for configuration management
Git - Source Code Version Control System

- Used to clone the source code repository

https://git-scm.com/
Vagrant - Software Development Environment

- Used to instantiate the virtual machine

https://www.vagrantup.com/
Vagrant OpenStack Cloud Provider

- Used by Vagrant to interface with OpenStack

https://github.com/ggiamarchi/vagrant-openstack-provider

- To install the plugin (after installing Vagrant):

  $ vagrant plugin install vagrant-openstack-provider
Ansible - Automation/Configuration Management

- Used to automate all VM provisioning and configuration management

https://www.ansible.com
What Exactly Does this Project do?

1. Instantiate a single virtual machine on OpenStack
2. Optional: Attach a 1 GB storage volume (Vagrant plugin)
3. Assign a static private IP address (Vagrant plugin)
4. Create a default user account (non-root system admin)
5. Configure a basic firewall (ingress ports 22, 80, 443)
6. Install a LEMP stack (Linux, NGINX, MariaDB, PHP)
7. Deploy SSL certs for HTTPS (Let’s Encrypt)
8. Install the WordPress web application
9. CentOS: Modify the webroot SELinux file context types (semanage)
Creating the OpenStack Storage Volume

Volumes

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ivm-814_volume_1GB</td>
<td>1 GB volume</td>
</tr>
</tbody>
</table>

Displaying 1 item
Creating Your SSH Key Pair

Access & Security

Key Pair Name
- yoga3pro

Displaying 1 item
Security Group - Default

Access & Security

<table>
<thead>
<tr>
<th>Security Groups</th>
<th>Key Pairs</th>
<th>Floating IPs</th>
<th>API Access</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>default</td>
<td>Default security group</td>
</tr>
</tbody>
</table>

Displaying 1 item
Security Group - Default - New Rules

<table>
<thead>
<tr>
<th>Ingress</th>
<th>IPv4</th>
<th>TCP</th>
<th>22 (SSH)</th>
<th>0.0.0.0/0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ingress</td>
<td>IPv4</td>
<td>TCP</td>
<td>80 (HTTP)</td>
<td>0.0.0.0/0</td>
</tr>
<tr>
<td>Ingress</td>
<td>IPv4</td>
<td>TCP</td>
<td>443 (HTTPS)</td>
<td>0.0.0.0/0</td>
</tr>
</tbody>
</table>
Floating IP

Access & Security

<table>
<thead>
<tr>
<th>Security Groups</th>
<th>Key Pairs</th>
<th>Floating IPs</th>
<th>API Access</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>IP Address</th>
<th>Mapped Fixed IP Addr</th>
</tr>
</thead>
<tbody>
<tr>
<td>206.12.59.178</td>
<td>-</td>
</tr>
</tbody>
</table>

Displaying 1 item
Finally, it’s time to start working with the actual code...
1st Action - Clone the Git Repository

- Create and then navigate to your development directory
  
  $ mkdir ~/Development
  
  $ cd ~/Development

- Clone the repository from GitHub
  
  $ git clone https://github.com/brentg71/openstack-wordpress.git

- Now navigate to the project’s main directory
  
  $ cd ~/Development/openstack-wordpress
2nd Action - Configure the OpenStack Environment

Configuration Part 1: `openstack_vars.rb`

- Copy the sample file to your home directory
  
  $ cp docs/openstack_vars.rb_SAMPLE ~/openstack_vars.rb

- Edit the contents to match your OpenStack environment
  
  $ vim ~/openstack_vars.rb

- Modify file permissions
  
  $ chmod 0600 ~/openstack_vars.rb
WARNING - openstack_vars.rb - WARNING

This file is required by the Vagrantfile.

Without this information, absolutely nothing in this project will work at all.
Where do I Find the OpenStack Env Info?

Access & Security

<table>
<thead>
<tr>
<th>Service</th>
<th>Service Endpoint</th>
</tr>
</thead>
<tbody>
<tr>
<td>Compute</td>
<td><a href="https://west.cloud.computecanada.ca">https://west.cloud.computecanada.ca</a>:</td>
</tr>
<tr>
<td>Network</td>
<td><a href="https://west.cloud.computecanada.ca">https://west.cloud.computecanada.ca</a>:</td>
</tr>
<tr>
<td>Volumev2</td>
<td><a href="https://west.cloud.computecanada.ca">https://west.cloud.computecanada.ca</a>:</td>
</tr>
</tbody>
</table>
Where Else do I Find the OpenStack Env Info?

- View or download your OpenStack credentials information by clicking one of these two buttons.
Configuration Part 2: Encrypted Variable Files

- 2 files in this repository which I have password encrypted using "ansible-vault"

ansible/group_vars/all

ansible/roles/users/files/default_id_rsa.pub
ansible/group_vars/all

- Contains information that I didn’t want to store in plain text in any Git repository
  - the root password for your VM (hash)
  - default user information
  - the mysql root password
  - the WordPress database name and user credentials info
ansible/group_vars/all cont...

- Copy the sample (unencrypted) "group_vars_all_sample" into the Ansible "group_vars" directory and rename it

  $ cp docs/group_vars_all_sample ansible/group_vars/all

- Edit this file and replace the sample information with your information

  $ vim ansible/group_vars/all

- Optional: encrypt this file using the "ansible-vault" command

  $ ansible-vault encrypt ansible/group_vars/all
Editing: ansible/group_vars/all

- At any point, if you need to edit this file again (after encrypting it)

  $ ansible-vault edit ansible/group_vars/all
For some reason, I decided to encrypt the SSH public key that I assigned to my default user

Copy your own SSH public key as follows

```
$ cp ~/.ssh/id_rsa.pub ansible/roles/users/files/default_id_rsa.pub
```

Optional: Encrypt your key once again using "ansible-vault"

```
$ ansible-vault encrypt ansible/roles/users/files/default_id_rsa.pub
```

Use the same password as you did for the previous file
Finally, we are now ready to execute the vaunted "single command"
Deploy and Provision

- Deploy and provision your virtual machine instance
  
  $ ./vagrant_up_OS.sh

- Basically, this script just executes the following command
  
  $ vagrant up --provider=openstack

This will take a while. The software updates alone could take a very long time. **Do not panic.** You will receive timely updates on your screen which will inform you about the current status.
[brentg@brentwg openstack-wordpress]$ ./vagrant_up_OS.sh
Bringing machine 'default' up with 'openstack' provider...

==> default: Finding flavor for server...
==> default: Finding image for server...
==> default: Finding network(s) for server...
==> default: Finding volume(s) to attach on server...
==> default: Launching a server with the following settings...
  *** default:  -- Tenant : ivm-814
  *** default:  -- Name   : wordpress1
  *** default:  -- Flavor : p1-1.5gb
Deploy and Provision - Ansible Output

PLAY [default] **********

TASK [setup] **********
ok: [default]

TASK [hostname] **********
changed: [default]

TASK [Create an ext4 file system on the new volume] **********
changed: [default]

TASK [Mount the nginx mount point on the new volume] **********
changed: [default]

TASK [Add hostname info to /etc/hosts file] **********
changed: [default]

TASK [YUM - Update all software packages] **********
changed: [default]
Deploy and Provision - Output Summary

TASK [wordpress : CentOS - Change the SELinux context for the NGINX webroot] ***
changed: [default]

TASK [wordpress : CentOS - Make SELinux context changes permanent] ****************
changed: [default]

PLAY RECAP ************************************************************
default : ok=76  changed=61  unreachable=0  failed=0
Connect Remotely, Reprovision, and Delete

- To connect remotely to the VM via ssh (as Vagrant’s default VM user)
  
  `$ vagrant ssh`

- (CAUTION) To provision new/edited Ansible role modifications to the VM
  
  `$ vagrant provision`

- (EVEN MORE CAUTION) To delete the VM
  
  `$ vagrant destroy`
Verifying Your Deployment

- Launch a web browser and navigate to your domain
  - You should be prompted to select your WordPress language
Next Steps

- Learn to use Otto instead of Vagrant for application deployment
- Use other plugins to deploy the application on other cloud platforms, such as DigitalOcean, AWS, Azure, and somehow VMware vCloud Director
- Perform actual “orchestration” (using OpenStack Heat)
- Use containers to deploy the application (using docker compose)
- Evolve and learn to use CoreOS, etcd, Fleet, Weave, Flannel, Heat, Magnum, and Kubernetes to deploy scalable microservices that automatically expand or contract as the demand for resources increase or decrease
Questions