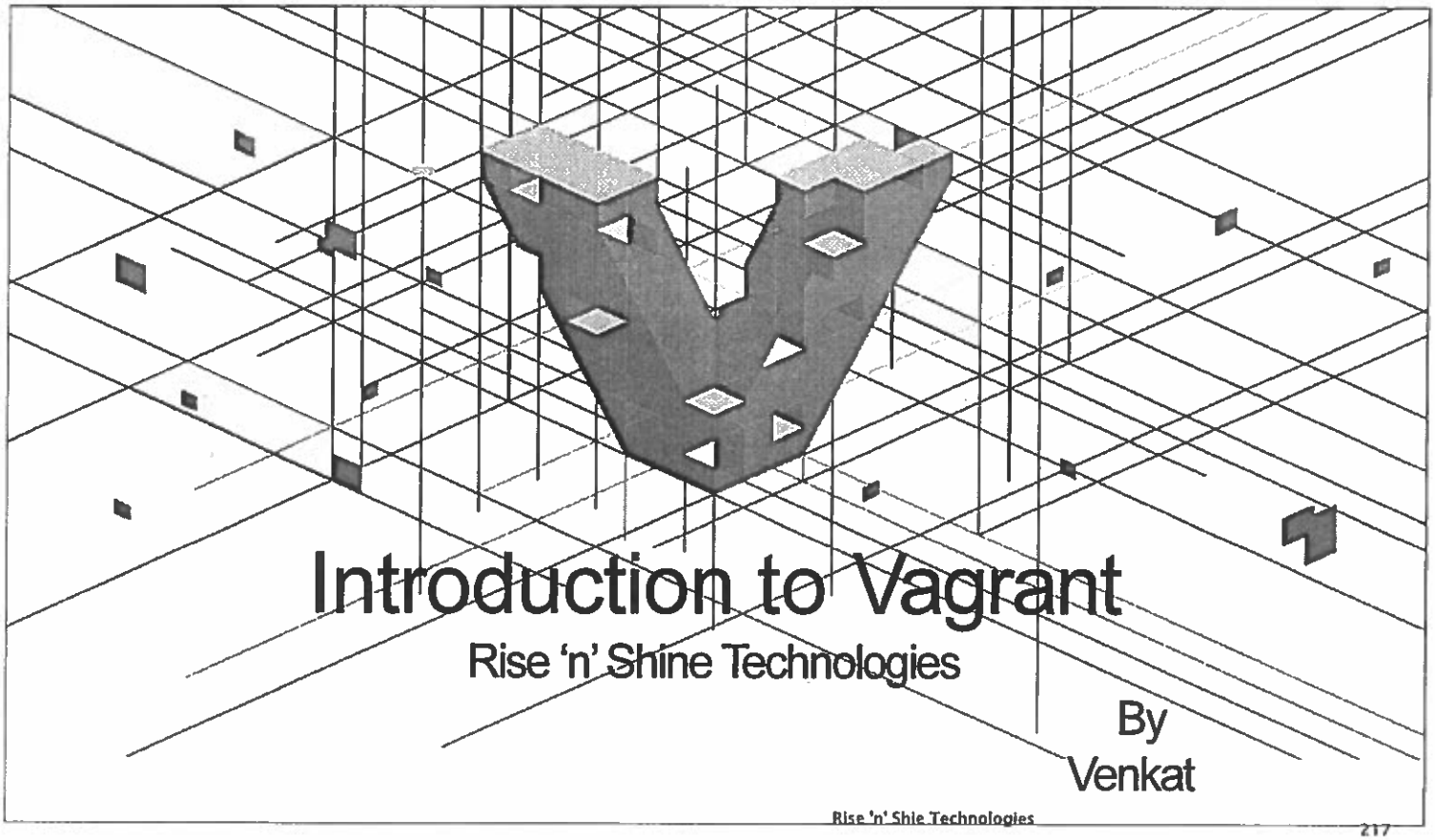
DevOps-6



Introduction to Vagrant

Rise 'n' Shine Technologies

By
Venkat

We're sorry, but something went wrong.

We've been notified about this issue and we'll take a look at it shortly.

What is Vagrant?

- A tool to build development environments based on virtual machines
- Focused to create environments that are similar as possible or identical with production servers
- Created by Mitchell Hashimoto
- Written in Ruby
- Initially builted on top of VirtualBox API, today offers VMWare Fusion support (as $79 per licence)

What is Vagrant?

Let Vagrant handle the entire lifecycle of development machines

- Suspend, Halt, Resume virtual machines
- Destroy your virtual machines and delete its metadata
- SSH in to your machine
- Package up your machines entire state and the re distribute it to other development team members
- Vagrant is a must have tool in development environments in which it will resemble your actual production

Why should we use Vagrant?

- Traditionally we would have to manually install and setup things like, Apache and MySQL locally on our development workstations.
- This can be very complex with all amount of moving parts to keep track of
- Human error is easy to take place when setting up complete local development environments
- Vagrant is an awesome way of fixing these issues by automating our setup of all these services needed to develop our applications. Each project gets its own virtual machines.

The Vagrant Workflow

- Developers can now check out versions or repositories from a version control such as GIT and run 'vagrant up".
- Now the developers can work on their own machines comfortably with their own, editors, browsers and tools.
- We have a consistent and stable development environment now
- The same scripts of configuration in using vagrant to deploy dev environments is the same used in production therefore the dev environment is as close as possible to the production limiting issues later on.

Prerequisites to Install Vagrant

- Tools Required:
  - VirtualBox
    - virtualization software
  - Vagrant
    - virtualization automation
- Steps:
  - 1. Download VirtualBox first
  - 2. Download Vagrant installer on Vagrant site (Debian, CentOS, Windows, OSX, other OS's)
  - 3. Get a Vagrant box

Prerequisites to Install Vagrant

- You can actually use Vagrant with other "providers" instead of just VirtualBox
  - VMware
  - AWS EC2
  - And just about any other provider out there!
- Note: In this course we are going to be primarily using VirtualBox because it's free.
- *Vagrant offer an official VMware provider for cost ww.virtualbox.com/vmware

Oracle VM VirtualBox

- Virtualization software from Oracle
- Free
- Runs on Windows, Mac, Linux
- Runs as an application
- Allows us to use local VMs
- Easy to install
- Works well with Vagrant

- https://www.virtualbox.org/

Vagrant

- Virtualization workflow software from HashiCorp
- Free, open-source
- Runs on Windows, Mac, Linux
- Easy to install
- Works well with Puppet, Chef, Shell
  - many other **provisioners**
- Works well with VirtualBox, VMware, Amazon Web Services
  - many other **providers**

- https://www.vagrantup.com/

What is a Vagrant Box?

- Vagrant boxes are just 'Templates'
- Is a previously builted Vagrant virtual machine image,
- ready-to-run
- Available in a lot of platforms (Linux, Windows)
- Manage Boxes such as:
  - Vagrant box list
  - Vagrant box add
  - Vagrant box remove

How to add a box?

- Great box repository:
  - www.vagrantbox.es
- Run this command:

```
$  vagrant --version
$ vagrant   box add <name>    <url>  <provider>   # virtualbox
```

- To create a VM:
  - Add the Vagrant box
  - $ vagrant box add hashicorp/precise64
    - downloads a "base box"
    - boxes at https://atlas.hashicorp.com/search

  - Inside your project, create a Vagrantfile:
    $ vagrant init hashicorp/precise64
    - builds a Vagrant file with the base box

Vagrantfile

- Vagrantfile
  - lots of comments by default
  - stock Vagrantfilewithout comments is:

  - Vagrant.configure(2) do |config| config.vm.box = "hashicorp/precise64"
  - End

  - As of date the current config version of vagrant is “2”
  - There are currently only 2 versions supported. “1” and “2”
  - Version “1” config files for Vagrant version 1.0.X
  - Version “2” config files for Vagrant version 1.1+ leading up to 2.0.x

Access Vagrant Box

- To access a VM:
  - vagrant ssh
    - connects to the VM via the forwarded SSH port

- requires an SSH client installed
  - Git (https://msysgit.github.io/)
  - openssh on Cygwin (http://www.cygwin.com/)
  - PuTTY (http://www.chiark.greenend.org.uk/~sgtatham/putty/)
    - requires converting the key format

- vagrant up
  - starts the VM with the following steps

- imports the base box to VirtualBox
- makes sure the base box is up to date
- sets a unique name for the VM
- sets up networking (just NAT by default)
- sets up port forwarding (just SSH by default)
- boots VM
- replaces known, insecure SSH key with a new random key
- makes sure VirtualBox Guest Additions are installed
- mounts shared folders (/vagrant by default on the VM)
- provisions software (nothing by default)

Rebuild Vagrant Box

- To rebuild a VM:
  - vagrant destroy
    - deletes a VM
  - vagrant up
    - starts the VM

How do start to use it?

Simply run this command:

```
$ vagrant up
```

Easy:

```
$ vagrant ssh
```

```
$ vagrant halt
```

```
$ vagrant reload
```

How to access it?

- You need to set forwarding ports between guest and host to work (bind on 0.0.0.0!)
- Just add the following code in your Vagrantfile, restart server and access in browser:

```
# -*- mode: ruby -*-
# vi: set ft=ruby :

Vagrant.configure("2") do |config|
  # ...
  config.vm.network :forwarded_port, guest: 3000,          host: 3000

  # ...
end
```

How to customize it?

- You can change memory, CPU cores and other things in Vagrantfile
- Just see VBoxManage options
- Example:

```
# -*- mode: ruby -*-
# vi: set ft=ruby :

Vagrant.configure("2") do |config|
  # ...
  config.vm.provider :virtualbox do |vb|
  vb.customize [ 'modifyvm', :id,      '--memory',    '1024' ]
  vb.customize [ 'modifyvm', :id,      '--cpus',      '4'    ]
  end

  # ...
end
```

[redacted]

•Of course, no! :)

•It's time to configure environment using available provisioners to install required software:

- •Shell

Using Shell

Create a single bash script that installs all you need:

```
#!/bin/bash

apt-get update

# base
apt-get install     --yes python nginx mongodb-server redis-server

# others
apt-get install     --yes curl tmux htop

(...)

# some additional configuration here

(...)
```

Easy:

```
$ vagrant   up --provision
```