Infrastructure as Code: Instant Development Environments

Instructor:

John J Rofrano

Senior Technical Staff Member, DevOps Champion
IBM T.J. Watson Research Center
rofrano@us.ibm.com (@JohnRofrano)

“DevOps Principle of the Day:

AUTOMATE EVERYTHING!”

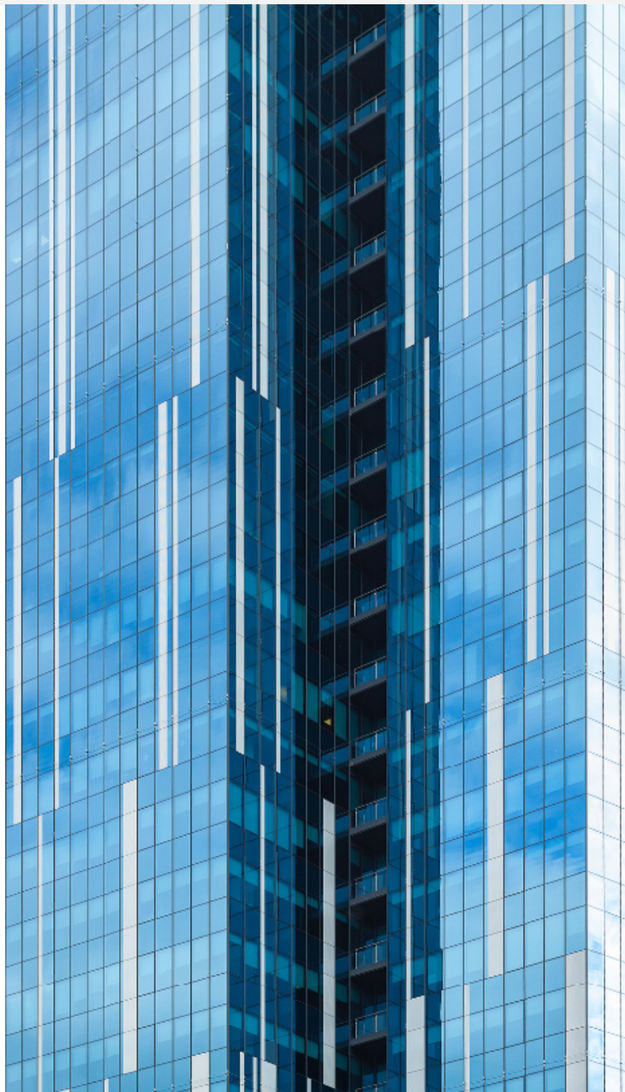
Even the Development Team !!!

What Will You Learn?

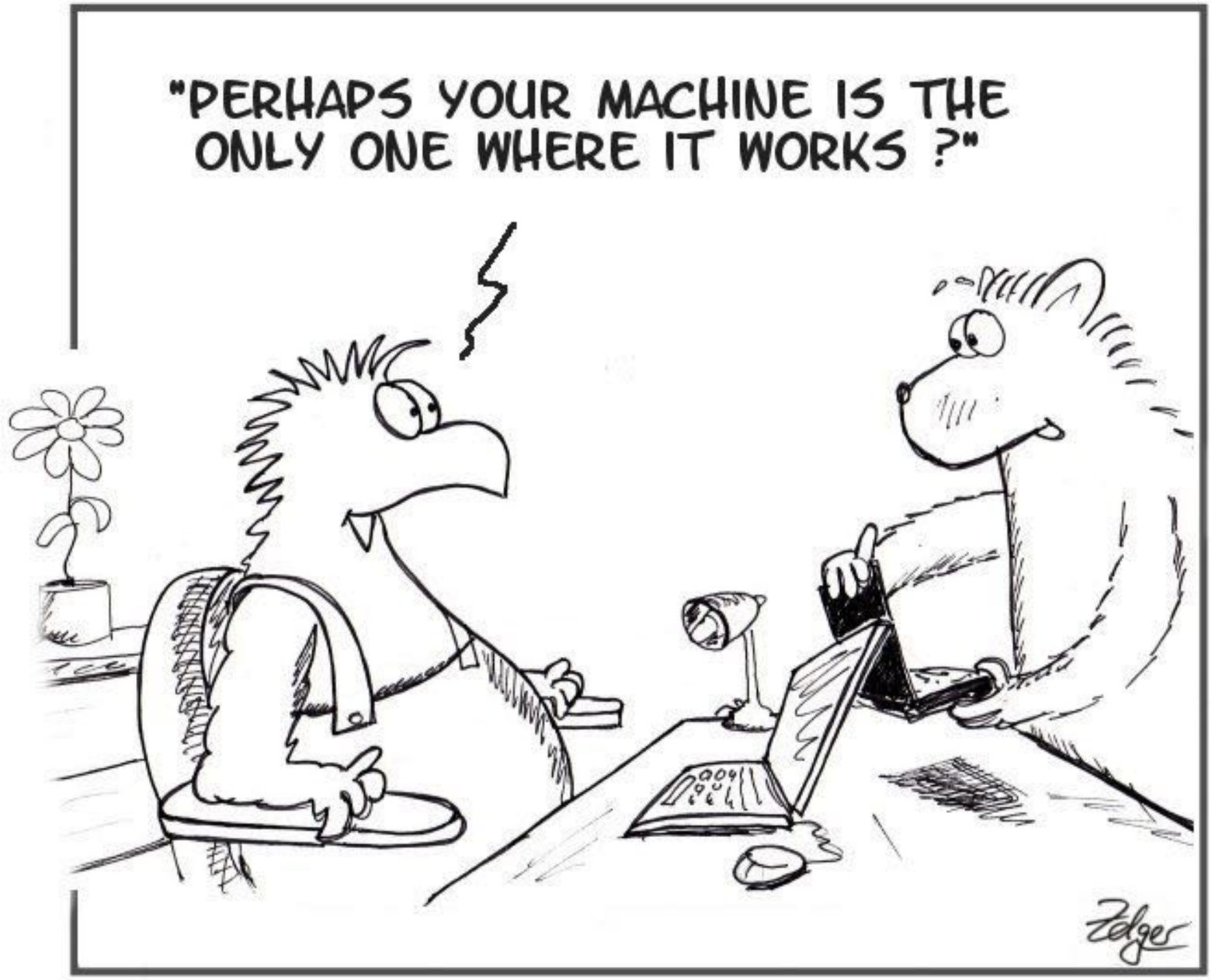
- How to create customized development environments with Vagrant and VirtualBox that can be created and destroyed and created again in a consistent fashion
- How to leverage Docker containers for middleware deployment
- How to make your DevOps team more productive with just a single command:> `vagrant up`

The Problem

- Manually creating local environments for developers to work in is:
  - Time consuming
  - Error prone
  - Inconsistent at best
  - Unreproducible at worst!



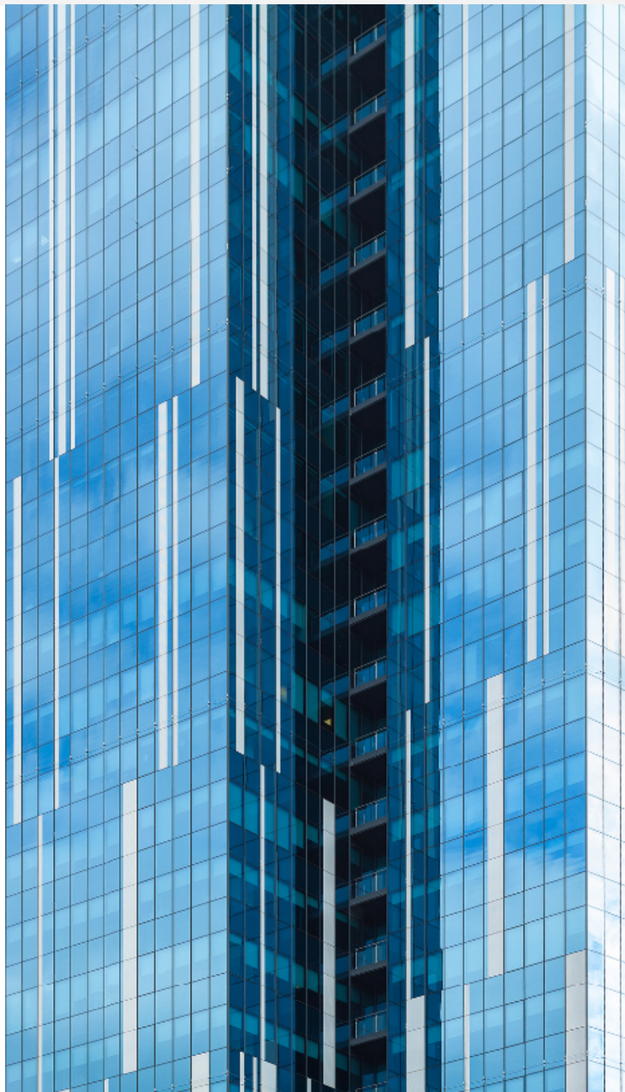
The Goal is to Avoid This



It works on my machine

The Wrong Solution

- Carefully and painstakingly create documentation in the form of “runbooks” that give step-by-step instructions on how to recreate the development environment via cut-n-paste
- NOT VERY AGILE !!!
  - We value: Working software over comprehensive documentation

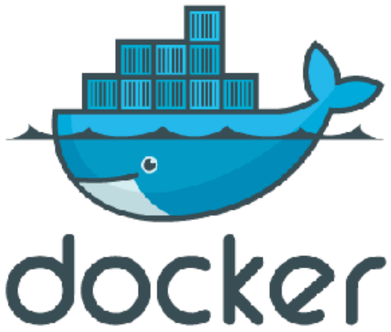


The Right Solution: Infrastructure as Code

Automate the creation of local development environments right on your laptop or desktop

Use Vagrant to quickly provision complex configurations consistently with repeatability

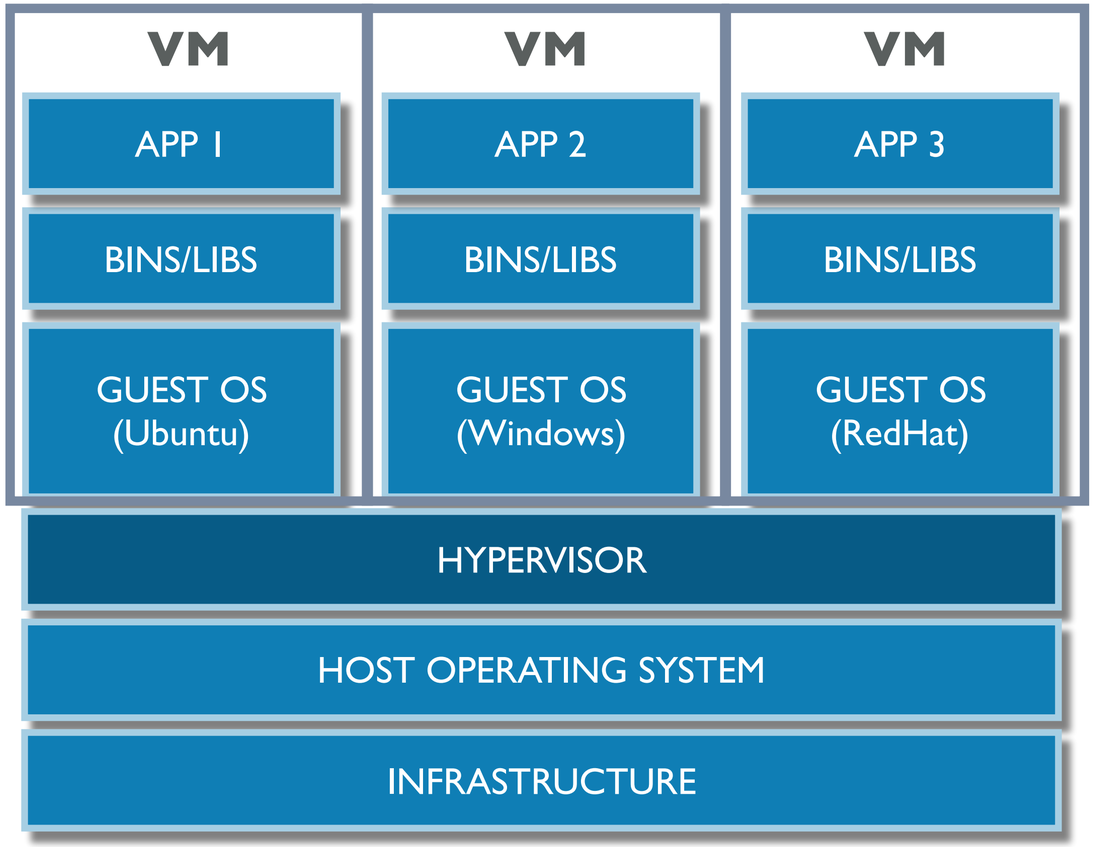
Use Docker to handle middleware without any installation



Use VirtualBox to create Virtual Machines to develop in

Virtual Machines

- **Virtual Machines** (VM) are created by emulating computer hardware in software
- The emulation is provided by software called a **Hypervisor**
- Each **Guest OS** thinks it's talking to dedicated computer hardware but it is really talking to the hypervisor that is sharing a much larger system



What is VirtualBox?

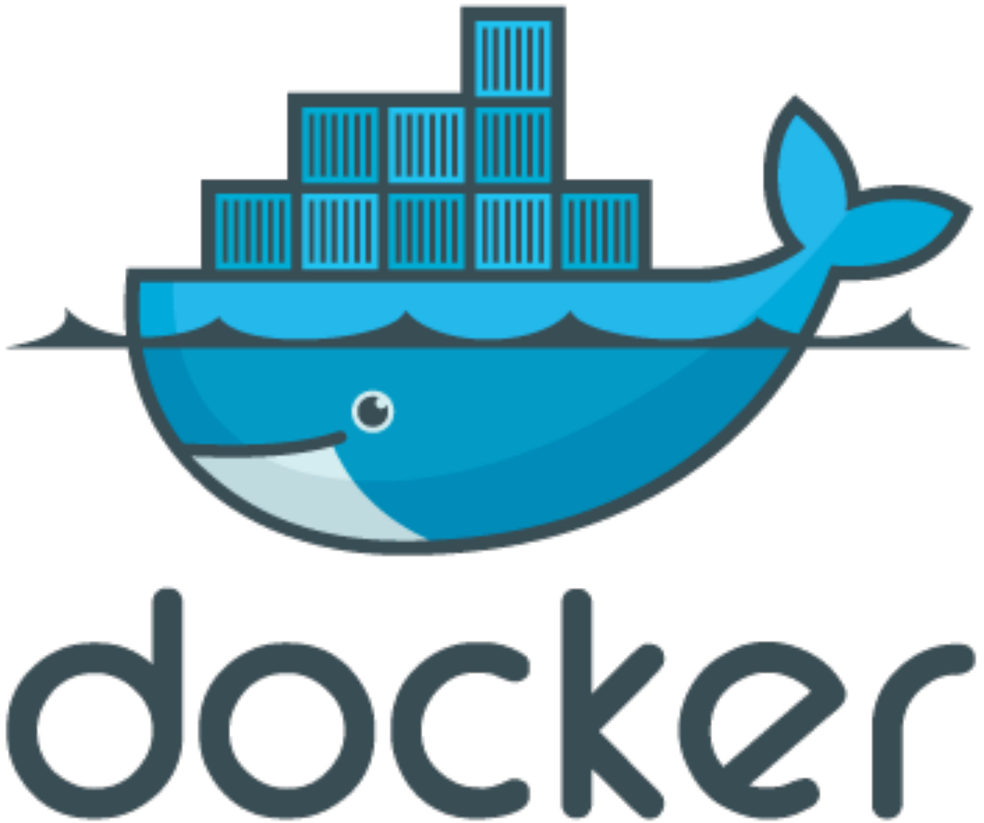
- VirtualBox is a free Hypervisor that runs on OS X, Windows, and Linux
- Similar to VMware Workstation on a PC, or VMware Fusions and Parallels Desktop on a Mac
- Allows you to run your code in a Virtual Machine

What is Vagrant?

- Vagrant is a developers tool for creating lightweight, reproducible and portable virtual environments via command-line
- It supports VirtualBox, VMware, SoftLayer, Amazon AWS and Digital Ocean
- It supports configuration management utilities like Puppet, Chef, etc.
- It supports Docker natively which makes is easy to use Docker containers in VMs

What is Docker?

- Docker packages applications into Containers that include all of their dependencies, but share the kernel with other containers
- Containers run as an isolated process in userspace on the host operating system
- Containers are not tied to any specific infrastructure
  - Docker containers run on any computer, on any infrastructure and in any cloud.
- This guarantees that it will always run the same, regardless of the environment it is running in.



Baked vs. Fried

- Do you Bake software into the image?
  - VM's can be created quicker because most software is pre-installed
- Or do you Fry it up fresh when you need it?
  - More versatile and less maintenance when software needs updating
- I like to Fry!


Summit of Innovation
Sea of Technical Debt

Hands-On

“live session”

Demo from Scratch

- Create a folder to work in:

```
$ mkdir lab-vagrant-demo
$ cd lab-vagrant-demo
```

- Go from Zero to running Ubuntu 18.04 Bionic 64 with these 2 commands:

```
$ vagrant init ubuntu/bionic64
$ vagrant up --provider virtualbox
```

- That’s it! 😀

* `--provider virtualbox` (only needs for Windows)

What Did “INIT” do?

- A default `Vagrantfile` was created with the `init` command
- The `Vagrantfile` controls how the VM will be provisioned
- By editing the `Vagrantfile` we can change the VM’s behavior
- The `Vagrantfile` will be what you check-in to GitHub

Default Vagrantfile

```
Vagrant.configure(2) do |config|

  config.vm.box = "ubuntu/bionic64"

  # config.vm.box_check_update = false

  # config.vm.network "forwarded_port", guest: 80, host: 8080

  # config.vm.network "private_network", ip: "192.168.33.10"

  # config.vm.network "public_network"

  # config.vm.synced_folder "../data", “/vagrant_data"

  # config.vm.provider "virtualbox" do |vb|
  #   vb.gui = true
  #   vb.memory = "1024"
  # end

  # config.push.define "atlas" do |push|
  #   push.app = "YOUR_ATLAS_USERNAME/YOUR_APPLICATION_NAME"
  # end

  # config.vm.provision "shell", inline: <<-SHELL
  #   apt-get update
  #   apt-get install -y apache2
  # SHELL
end
```

Default Vagrantfile

This is a Ruby file that uses Ruby syntax

```
Vagrant.configure(2) do |config|

  config.vm.box = "ubuntu/bionic64"

  # config.vm.box_check_update = false

  # config.vm.network "forwarded_port", guest: 80, host: 8080

  # config.vm.network "private_network", ip: "192.168.33.10"

  # config.vm.network "public_network"

  # config.vm.synced_folder "../data", “/vagrant_data"

  # config.vm.provider "virtualbox" do |vb|
  #   vb.gui = true
  #   vb.memory = "1024"
  # end

  # config.push.define "atlas" do |push|
  #   push.app = "YOUR_ATLAS_USERNAME/YOUR_APPLICATION_NAME"
  # end

  # config.vm.provision "shell", inline: <<-SHELL
  #   apt-get update
  #   apt-get install -y apache2
  # SHELL
end
```

Default Vagrantfile

```
Vagrant.configure(2) do |config|

  config.vm.box = "ubuntu/bionic64"

  # config.vm.box_check_update = false

  # config.vm.network "forwarded_port", guest: 80, host: 8080

  # config.vm.network "private_network", ip: "192.168.33.10"

  # config.vm.network "public_network"

  # config.vm.synced_folder "../data", “/vagrant_data"

  # config.vm.provider "virtualbox" do |vb|
  #   vb.gui = true
  #   vb.memory = "1024"
  # end

  # config.push.define "atlas" do |push|
  #   push.app = "YOUR_ATLAS_USERNAME/YOUR_APPLICATION_NAME"
  # end

  # config.vm.provision "shell", inline: <<-SHELL
  #   apt-get update
  #   apt-get install -y apache2
  # SHELL
end
```

This is a Ruby file that uses Ruby syntax

Selects the box to use as a base OS

Default Vagrantfile

```
Vagrant.configure(2) do |config|

  config.vm.box = "ubuntu/bionic64"

  # config.vm.box_check_update = false

  # config.vm.network "forwarded_port", guest: 80, host: 8080

  # config.vm.network "private_network", ip: "192.168.33.10"

  # config.vm.network "public_network"

  # config.vm.synced_folder "../data", “/vagrant_data"

  # config.vm.provider "virtualbox" do |vb|
  #   vb.gui = true
  #   vb.memory = "1024"
  # end

  # config.push.define "atlas" do |push|
  #   push.app = "YOUR_ATLAS_USERNAME/YOUR_APPLICATION_NAME"
  # end

  # config.vm.provision "shell", inline: <<-SHELL
  #   apt-get update
  #   apt-get install -y apache2
  # SHELL
end
```

This is a Ruby file that uses Ruby syntax

Selects the box to use as a base OS

Everything else is commented out

What Did “UP” Do?

- Vagrant downloaded the ubuntu/bionic64 box from atlas.hashicorp.com
- VirtualBox was called to create a VM from that box
- Installed VirtualBox tools and setup the network and shared your current folder as /vagrant
- Ran provisioning scripts to prepare the VM (more on that later)
- Set up SSH keys to logon to the VM

Initial Vagrant Run

```
$ vagrant up
Bringing machine 'default' up with 'virtualbox' provider...
==> default: Checking if box 'ubuntu/bionic64' is up to date...
==> default: Clearing any previously set forwarded ports...
==> default: Clearing any previously set network interfaces...
==> default: Preparing network interfaces based on configuration...
    default: Adapter 1: nat
    default: Adapter 2: hostonly
==> default: Forwarding ports...
    default: 5000 (guest) => 5000 (host) (adapter 1)
    default: 22 (guest) => 2222 (host) (adapter 1)
==> default: Running 'pre-boot' VM customizations...
==> default: Booting VM...
==> default: Waiting for machine to boot. This may take a few minutes...
    default: SSH address: 127.0.0.1:2222
    default: SSH username: ubuntu
    default: SSH auth method: password
    default: Warning: Remote connection disconnect. Retrying...
    default: Warning: Connection reset. Retrying...
==> default: Machine booted and ready!
[default] GuestAdditions 5.1.26 running --- OK.
==> default: Checking for guest additions in VM...
==> default: Configuring and enabling network interfaces...
==> default: Mounting shared folders...
    default: /vagrant => /Users/rofrano/github/lab-vagrant
==> default: Machine already provisioned. Run `vagrant provision` or use the `--provision`
==> default: flag to force provisioning. Provisioners marked to run always will still run.
```

Initial Vagrant Run

```
$ vagrant up
Bringing machine 'default' up with 'virtualbox' provider...
==> default: Checking if box 'ubuntu/bionic64' is up to date...
==> default: Clearing any previously set forwarded ports...
==> default: Clearing any previously set network interfaces...
==> default: Preparing network interfaces based on configuration...
    default: Adapter 1: nat
    default: Adapter 2: hostonly
==> default: Forwarding ports...
    default: 5000 (guest) => 5000 (host) (adapter 1)
    default: 22 (guest) => 2222 (host) (adapter 1)
==> default: Running 'pre-boot' VM customizations...
==> default: Booting VM...
==> default: Waiting for machine to boot. This may take a few minutes...
    default: SSH address: 127.0.0.1:2222
    default: SSH username: ubuntu
    default: SSH auth method: password
    default: Warning: Remote connection disconnect. Retrying...
    default: Warning: Connection reset. Retrying...
==> default: Machine booted and ready!
[default] GuestAdditions 5.1.26 running --- OK.
==> default: Checking for guest additions in VM...
==> default: Configuring and enabling network interfaces...
==> default: Mounting shared folders...
    default: /vagrant => /Users/rofrano/github/lab-vagrant
==> default: Machine already provisioned. Run `vagrant provision` or use the `--provision`
==> default: flag to force provisioning. Provisioners marked to run always will still run.
```

Creates the VM

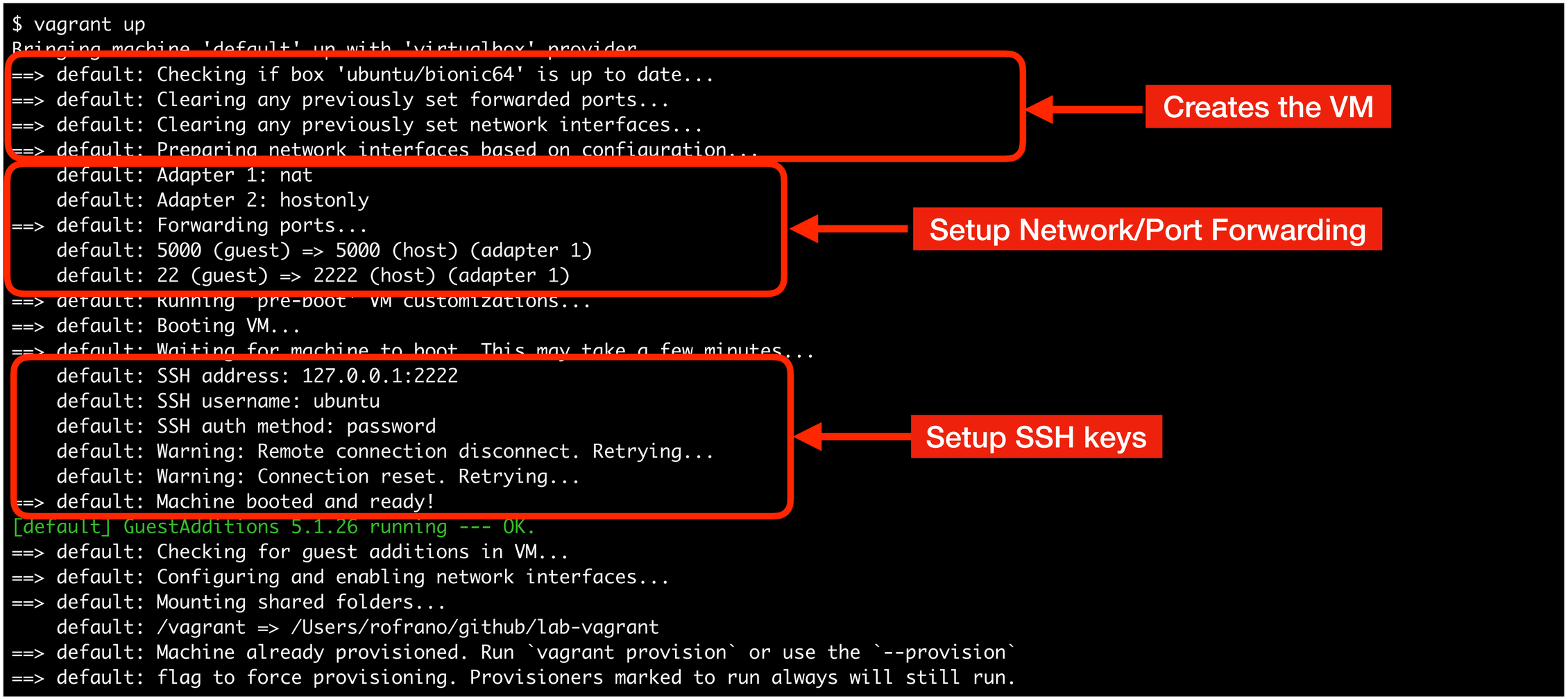
Initial Vagrant Run

```
$ vagrant up
Bringing machine 'default' up with 'virtualbox' provider...
==> default: Checking if box 'ubuntu/bionic64' is up to date...
==> default: Clearing any previously set forwarded ports...
==> default: Clearing any previously set network interfaces...
==> default: Preparing network interfaces based on configuration...
    default: Adapter 1: nat
    default: Adapter 2: hostonly
==> default: Forwarding ports...
    default: 5000 (guest) => 5000 (host) (adapter 1)
    default: 22 (guest) => 2222 (host) (adapter 1)
==> default: Running `pre-boot` VM customizations...
==> default: Booting VM...
==> default: Waiting for machine to boot. This may take a few minutes...
    default: SSH address: 127.0.0.1:2222
    default: SSH username: ubuntu
    default: SSH auth method: password
    default: Warning: Remote connection disconnect. Retrying...
    default: Warning: Connection reset. Retrying...
==> default: Machine booted and ready!
[default] GuestAdditions 5.1.26 running --- OK.
==> default: Checking for guest additions in VM...
==> default: Configuring and enabling network interfaces...
==> default: Mounting shared folders...
    default: /vagrant => /Users/rofrano/github/lab-vagrant
==> default: Machine already provisioned. Run `vagrant provision` or use the `--provision`
==> default: flag to force provisioning. Provisioners marked to run always will still run.
```

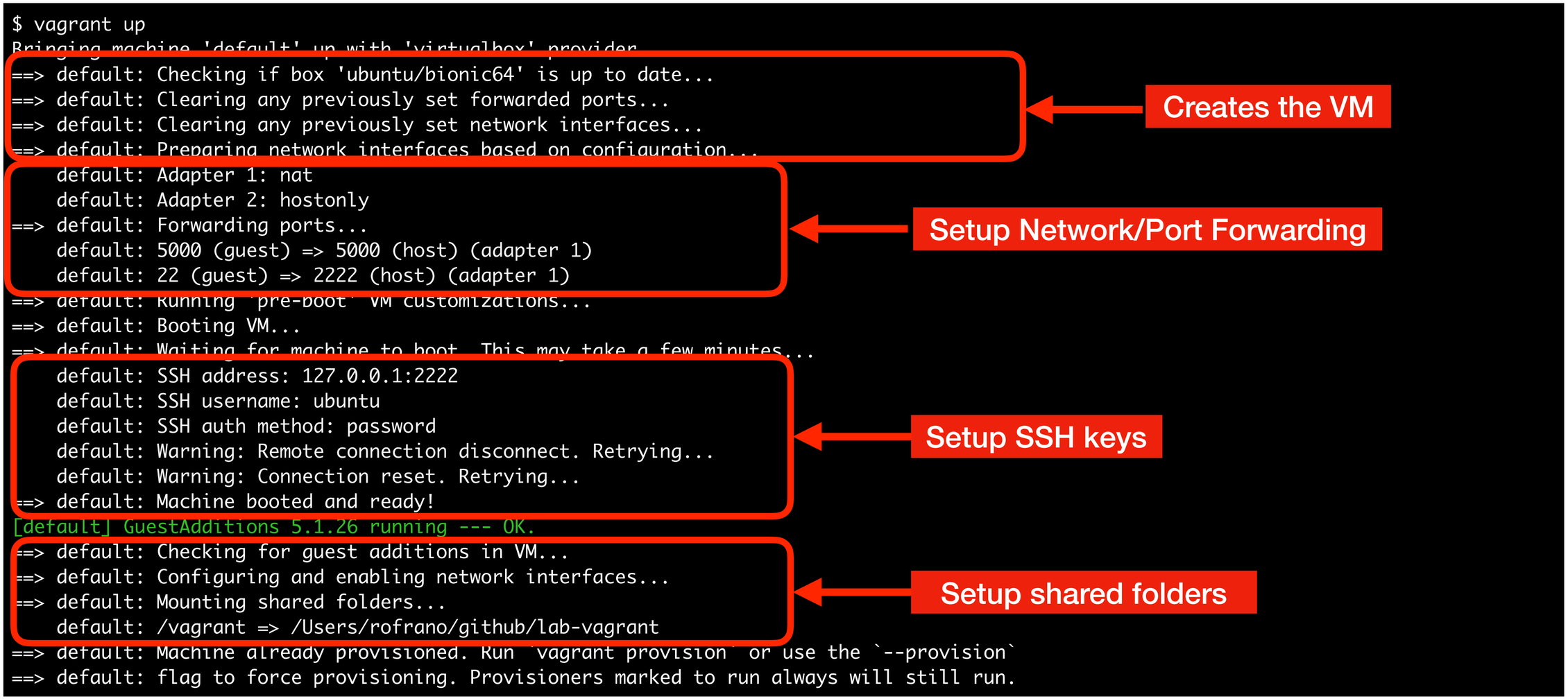
Creates the VM

Setup Network/Port Forwarding

Initial Vagrant Run

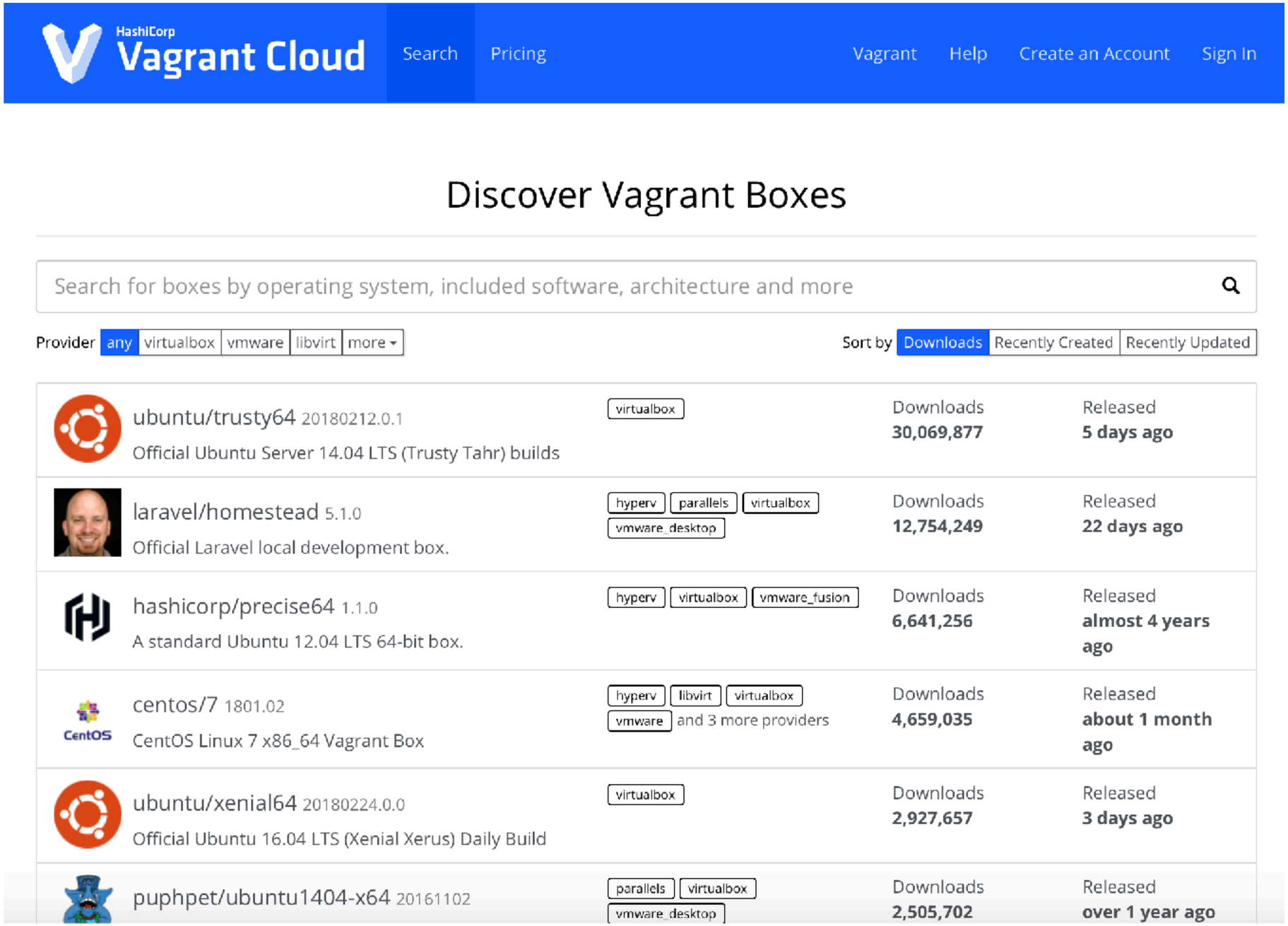


Initial Vagrant Run



Where did Ubuntu/ Bionic64 Come From?

https://app.vagrantup.com/boxes/search



What Other Boxes Are There?

http://www.vagrantbox.es/

Vagrantbox.es

Fork me on GitHub

Vagrant is an amazing tool for managing virtual machines via a simple to use command line interface. With a simple `vagrant up` you can be working in a clean environment based on a standard template.

These standard templates are called base boxes, and this website is simply a list of boxes people have been nice enough to make publicly available.

Suggest a Box

Do you know of another base box? Send a pull request and we'll add it to the list below.

Available Boxes

To use the available boxes just replace {title} and {url} with the information in the table below.

```
$ vagrant box add {title} {url}
$ vagrant init {title}
$ vagrant up
```

Search:

Name	Provider	URL	Size (MB)
Virtuozzo 7 x64 (Guest Additions 4.3.26)	VirtualBox	Copy https://atlas.hashicorp.com/OpenVZ/boxes/Virtuozzo-7b2	912
Debian Jessie 8.1.0 Release x64 (Minimal, Guest Additions 4.3.26)	VirtualBox	Copy https://atlas.hashicorp.com/ARTACK/boxes/debian-jessie	692
CentOS 7.0 x64 (Minimal, VirtualBox Guest Additions 4.3.28, Puppet 3.8.1 - see here for more infos)	VirtualBox	Copy https://github.com/tommy-muehle/puppet-vagrant-boxes/releases/download/1.1.0/centos-7.0-x86_64.box	475
CentOS 6.6 x64 (Minimal, VirtualBox Guest Additions, Puppet 3.7.5 - see here for more infos)	VirtualBox	Copy https://github.com/tommy-muehle/puppet-vagrant-boxes/releases/download/1.0.0/centos-6.6-x86_64.box	348
CentOS 7 x64 (Minimal, Shrinked, Guest Additions 4.3.26) (Monthly updates)	VirtualBox	Copy https://github.com/holms/vagrant-centos7-box/releases/download/7.1.1503.001/CentOS-7.1.1503-x86_64-netboot.box	437
CentOS 7.1 x64 (Minimal, Puppet 4.2.3, Guest Additions 4.3.30)[notes]	VirtualBox	Copy https://github.com/CommanderK5/packer-centos-template/releases/download/0.7.1/vagrant-centos-7.1.box	576
CentOS 6.7 x64 (Minimal, Puppet 4.2.3, Guest Additions 4.3.30)[notes]	VirtualBox	Copy https://github.com/CommanderK5/packer-centos-template/releases/download/0.6.7/vagrant-centos-6.7.box	577
Debian Jessie 8.0 Release x64 (Minimal, Shrinked, Guest Additions 4.3.26)	VirtualBox	Copy https://github.com/holms/vagrant-jessie-box/releases/download/Jessie-v0.1/Debian-jessie-amd64-netboot.box	437
Debian 8.1 x64 (LXC, Puppet 3.7.2, Guest Additions 4.3.28)	VirtualBox	Copy https://dl.dropboxusercontent.com/u/3523744/boxes/debian-8.1-amd64-lxc-puppet/debian-8.1-lxc-puppet.box	594
Debian Jessie 8.0 RC2 64-bit (Mininimal, Shrinked + Guest Additions 4.3.24 + Puppet 3.7.2)	VirtualBox	Copy http://static.gender-api.com/debian-8-jessie-rc2-x64-slim.box	328

What Other Boxes Are There?

http://www.vagrantbox.es/

Vagrantbox.es

Vagrant is an amazing tool for managing virtual machines via a simple to use command line interface. With a simple `vagrant up` you can be working in a clean environment based on a standard template.

These standard templates are called base boxes, and this website is simply a list of boxes people have been nice enough to make publicly available.

Suggest a Box

Do you know of another base box? Send a pull request and we'll add it to the list below.

Available Boxes

To use the available boxes just replace {title} and {url} with the information in

```
$ vagrant box add {title} {url}
$ vagrant init {title}
$ vagrant up
```

DO NOT manually download new boxes ! ! !
It is better to specify them in your Vagrantfile
so that all developers have the same boxes

Search:

Name	Provider	URL	Size (MB)
Virtuozzo 7 x64 (Guest Additions 4.3.26)	VirtualBox	Copy https://atlas.hashicorp.com/OpenVZ/boxes/Virtuozzo-7b2	912
Debian Jessie 8.1.0 Release x64 (Minimal, Guest Additions 4.3.26)	VirtualBox	Copy https://atlas.hashicorp.com/ARTACK/boxes/debian-jessie	692
CentOS 7.0 x64 (Minimal, VirtualBox Guest Additions 4.3.28, Puppet 3.8.1 - see here for more infos)	VirtualBox	Copy https://github.com/tommy-muehle/puppet-vagrant-boxes/releases/download/1.1.0/centos-7.0-x86_64.box	475
CentOS 6.6 x64 (Minimal, VirtualBox Guest Additions, Puppet 3.7.5 - see here for more infos)	VirtualBox	Copy https://github.com/tommy-muehle/puppet-vagrant-boxes/releases/download/1.0.0/centos-6.6-x86_64.box	348
CentOS 7 x64 (Minimal, Shrinked, Guest Additions 4.3.26) (Monthly updates)	VirtualBox	Copy https://github.com/holms/vagrant-centos7-box/releases/download/7.1.1503.001/CentOS-7.1.1503-x86_64-netboot.box	437
CentOS 7.1 x64 (Minimal, Puppet 4.2.3, Guest Additions 4.3.30)[notes]	VirtualBox	Copy https://github.com/CommanderK5/packer-centos-template/releases/download/0.7.1/vagrant-centos-7.1.box	576
CentOS 6.7 x64 (Minimal, Puppet 4.2.3, Guest Additions 4.3.30)[notes]	VirtualBox	Copy https://github.com/CommanderK5/packer-centos-template/releases/download/0.6.7/vagrant-centos-6.7.box	577
Debian Jessie 8.0 Release x64 (Minimal, Shrinked, Guest Additions 4.3.26)	VirtualBox	Copy https://github.com/holms/vagrant-jessie-box/releases/download/Jessie-v0.1/Debian-jessie-amd64-netboot.box	437
Debian 8.1 x64 (LXC, Puppet 3.7.2, Guest Additions 4.3.28)	VirtualBox	Copy https://dl.dropboxusercontent.com/u/3523744/boxes/debian-8.1-amd64-lxc-puppet/debian-8.1-lxc-puppet.box	594
Debian Jessie 8.0 RC2 64-bit (Mininimal, Shrinked + Guest Additions 4.3.24 + Puppet 3.7.2)	VirtualBox	Copy http://static.gender-api.com/debian-8-jessie-rc2-x64-slim.box	328

How To Use A Different Box?

- To use a different box, specify a name in `config.vm.box` and the url in `config.vm.box_url`

```
Vagrant.configure(2) do |config|

 config.vm.box = "centos64-x86_64"
 config.vm.box_url = "https://github.com/2creatives/vagrant-centos/releases/
download/v6.4.2/centos64-x86_64-20140116.box"
 config.vm.box_download_insecure = true

end
```

Accessing Your VM

- Use SSH client to access your VM just as if it were a remote server

```
$ vagrant ssh
```

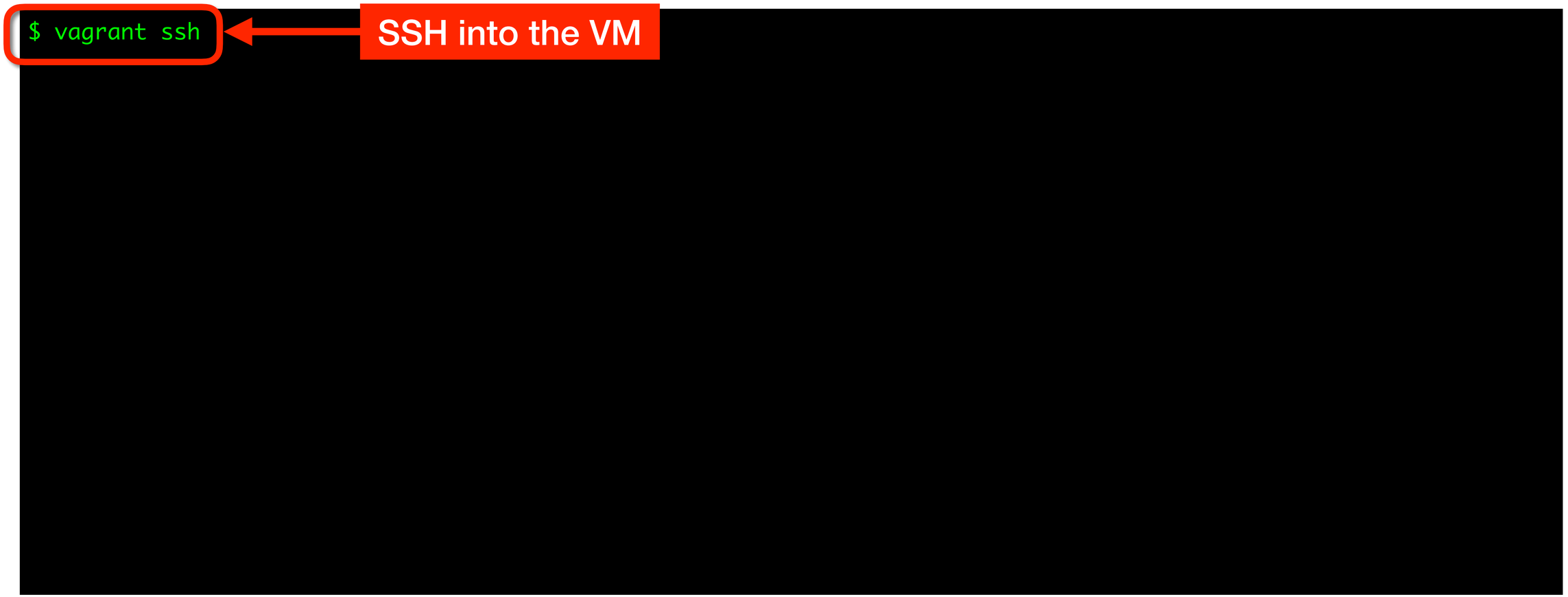
- The `/vagrant` folder holds your current directory on the host OS

```
$ cd /vagrant
```

Vagrant SSH

```
$ vagrant ssh
```

Vagrant SSH



Vagrant SSH

```
$ vagrant ssh

Welcome to Ubuntu 18.04.2 LTS (GNU/Linux 4.15.0-54-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

  Get cloud support with Ubuntu Advantage Cloud Guest:
    http://www.ubuntu.com/business/services/cloud

2 packages can be updated.
0 updates are security updates.


Last login: Wed Jul  3 15:19:31 2019 from 10.0.2.2
ubuntu@ubuntu-xenial:~$
```

Vagrant SSH

```
$ vagrant ssh

Welcome to Ubuntu 18.04.2 LTS (GNU/Linux 4.15.0-54-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

  Get cloud support with Ubuntu Advantage Cloud Guest:
    http://www.ubuntu.com/business/services/cloud

2 packages can be updated.
0 updates are security updates.


Last login: Wed Jul  3 15:19:31 2019 from 10.0.2.2
ubuntu@ubuntu-xenial:~$ cd /vagrant
ubuntu@ubuntu-xenial:/vagrant$ ls -l
```

Vagrant SSH

```
$ vagrant ssh

Welcome to Ubuntu 18.04.2 LTS (GNU/Linux 4.15.0-54-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

  Get cloud support with Ubuntu Advantage Cloud Guest:
    http://www.ubuntu.com/business/services/cloud

2 packages can be updated.
0 updates are security updates.


Last login: Wed Jul  3 15:19:31 2019 from 10.0.2.2
ubuntu@ubuntu-xenial:~$ cd /vagrant
ubuntu@ubuntu-xenial:/vagrant$ ls -l
```

access to local filesystem via the `/vagrant` folder

Vagrant SSH

```
$ vagrant ssh

Welcome to Ubuntu 18.04.2 LTS (GNU/Linux 4.15.0-54-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

  Get cloud support with Ubuntu Advantage Cloud Guest:
    http://www.ubuntu.com/business/services/cloud

2 packages can be updated.
0 updates are security updates.


Last login: Wed Jul  3 15:19:31 2019 from 10.0.2.2
ubuntu@ubuntu-xenial:~$ cd /vagrant
ubuntu@ubuntu-xenial:/vagrant$ ls -l
-rw-r--r-- 1 ubuntu ubuntu  4715 Sep  7 12:52 Vagrantfile
ubuntu@ubuntu-xenial:/vagrant$
```

Vagrant SSH

```
$ vagrant ssh

Welcome to Ubuntu 18.04.2 LTS (GNU/Linux 4.15.0-54-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

  Get cloud support with Ubuntu Advantage Cloud Guest:
    http://www.ubuntu.com/business/services/cloud

2 packages can be updated.
0 updates are security updates.


Last login: Wed Jul  3 15:19:31 2019 from 10.0.2.2
ubuntu@ubuntu-xenial:~$ cd /vagrant
ubuntu@ubuntu-xenial:/vagrant$ ls -l
-rw-r--r-- 1 ubuntu ubuntu  4715 Sep  7 12:52 Vagrantfile
ubuntu@ubuntu-xenial:/vagrant$
```

This is your `Vagrantfile` from inside the VM

Vagrant file sharing

- Vagrant is sharing the host folder it was invoked from within the vm as `/vagrant`
- This means that you and edit files with your native Mac or Windows editor
- ...and still be able to run the code on your Linux VM

Edit the Vagrantfile

- We can install software using Chef, Ansible, Puppet, Salt, Shell, etc.
- Let’s use the “shell” provisioner to add apache2

```
Vagrant.configure(2) do |config|

  config.vm.box = "ubuntu/bionic64"

  config.vm.provision "shell", inline: <<-SHELL
    apt-get update
    apt-get install -y apache2
  SHELL

end
```

Edit the Vagrantfile

- We can install software using Chef, Ansible, Puppet, Salt, Shell, etc.
- Let’s use the “shell” provisioner to add apache2

```
Vagrant.configure(2) do |config|

  config.vm.box = "ubuntu/bionic64"

  config.vm.provision "shell", inline: <<-SHELL
    apt-get update
    apt-get install -y apache2
  SHELL

end
```

Uncomment this code
from the bottom of the Vagrantfile
It uses shell syntax for the
OS (Ubuntu in this case)

Re-provision the VM after Updates

- After making changes to a `Vagrantfile` in any of the provisioning sections, you must re-provision the VM
- Exit and run the new provision block

```
$ exit

$ vagrant provision
```

Testing the Web Server

- Use SSH client to access your VM and CURL to access Apache

```
$ vagrant ssh
$ curl http://127.0.0.1
$ exit
```

- You should see HTML returned
- But if you access it from your browser as `127.0.0.1` it will fail! Why?

Let’s Forward The Web Port

- We must forward the ports that we want to access from outside the VM. Let’s add a private IP Address for direct access too:

```
Vagrant.configure(2) do |config|

  config.vm.box = “ubuntu/xenial64"

  # accessing "localhost:8080" will access port 80 on the guest machine.
  config.vm.network "forwarded_port", guest: 80, host: 8080

  config.vm.network "private_network", ip: "192.168.33.10"

  config.vm.provision "shell", inline: <<-SHELL
    sudo apt-get update
    sudo apt-get install -y apache2
  SHELL

end
```

Let’s Forward The Web Port

- We must forward the ports that we want to access from outside the VM. Let’s add a private IP Address for direct access too:

```
Vagrant.configure(2) do |config|

  config.vm.box = “ubuntu/xenial64"

  # accessing "localhost:8080" will access port 80 on the guest machine.
  config.vm.network "forwarded_port", guest: 80, host: 8080

  config.vm.network "private_network", ip: "192.168.33.10"

  config.vm.provision "shell", inline: <<-SHELL
    sudo apt-get update
    sudo apt-get install -y apache2
  SHELL

end
```

We should use ports higher than 1024 if we are not root

Reload the VM after Updates

- Since we didn't change any of the provisioning blocks in the `Vagrantfile`, we can just exit and reload the VM and the port forwarding will be picked up

```
$ vagrant reload
```

New Vagrant Run

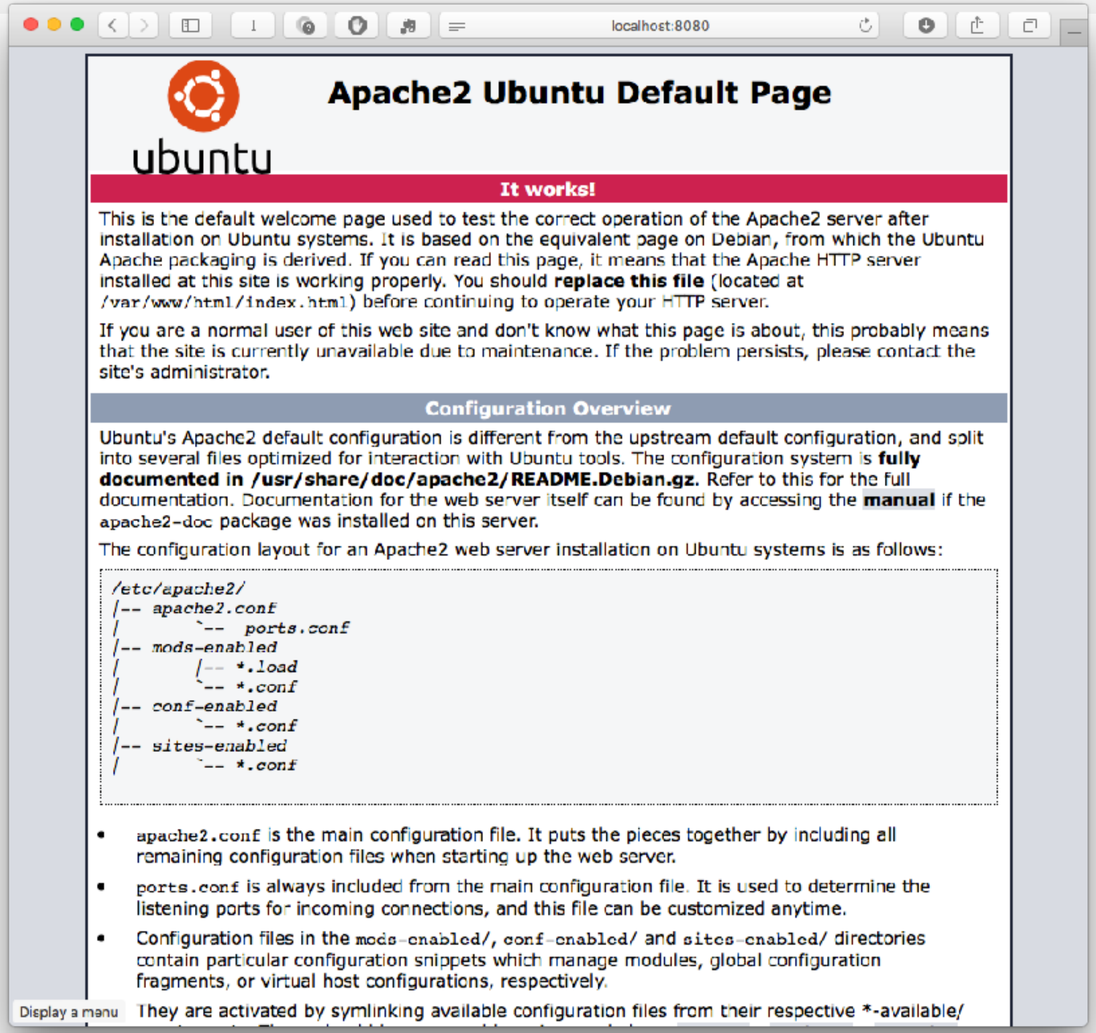
```
iotia:lab-vagrant-demo rofrano$ vagrant reload
==> default: Attempting graceful shutdown of VM...
==> default: Checking if box 'ubuntu/bionic64' is up to date...
==> default: Clearing any previously set forwarded ports...
==> default: Couldn't find Cheffile at ./Cheffile.
==> default: Clearing any previously set network interfaces...
==> default: Preparing network interfaces based on configuration...
    default: Adapter 1: nat
    default: Adapter 2: hostonly
==> default: Forwarding ports...
    default: 80 (guest) => 8080 (host) (adapter 1)
    default: 22 (guest) => 2222 (host) (adapter 1)
==> default: Booting VM...
==> default: Waiting for machine to boot. This may take a few minutes...
    default: SSH address: 127.0.0.1:2222
    default: SSH username: ubuntu
    default: SSH auth method: password
    default: Warning: Remote connection disconnect. Retrying...
    default: Warning: Remote connection disconnect. Retrying...
==> default: Machine booted and ready!
GuestAdditions 5.0.20 running --- OK.
==> default: Checking for guest additions in VM...
==> default: Configuring and enabling network interfaces...
==> default: Mounting shared folders...
    default: /vagrant => /Users/rofrano/github/Demo/lab-vagrant-demo
==> default: Machine already provisioned. Run `vagrant provision` or use the `--provision`
==> default: flag to force provisioning. Provisioners marked to run always will still run.
iotia:lab-vagrant-demo rofrano$
```

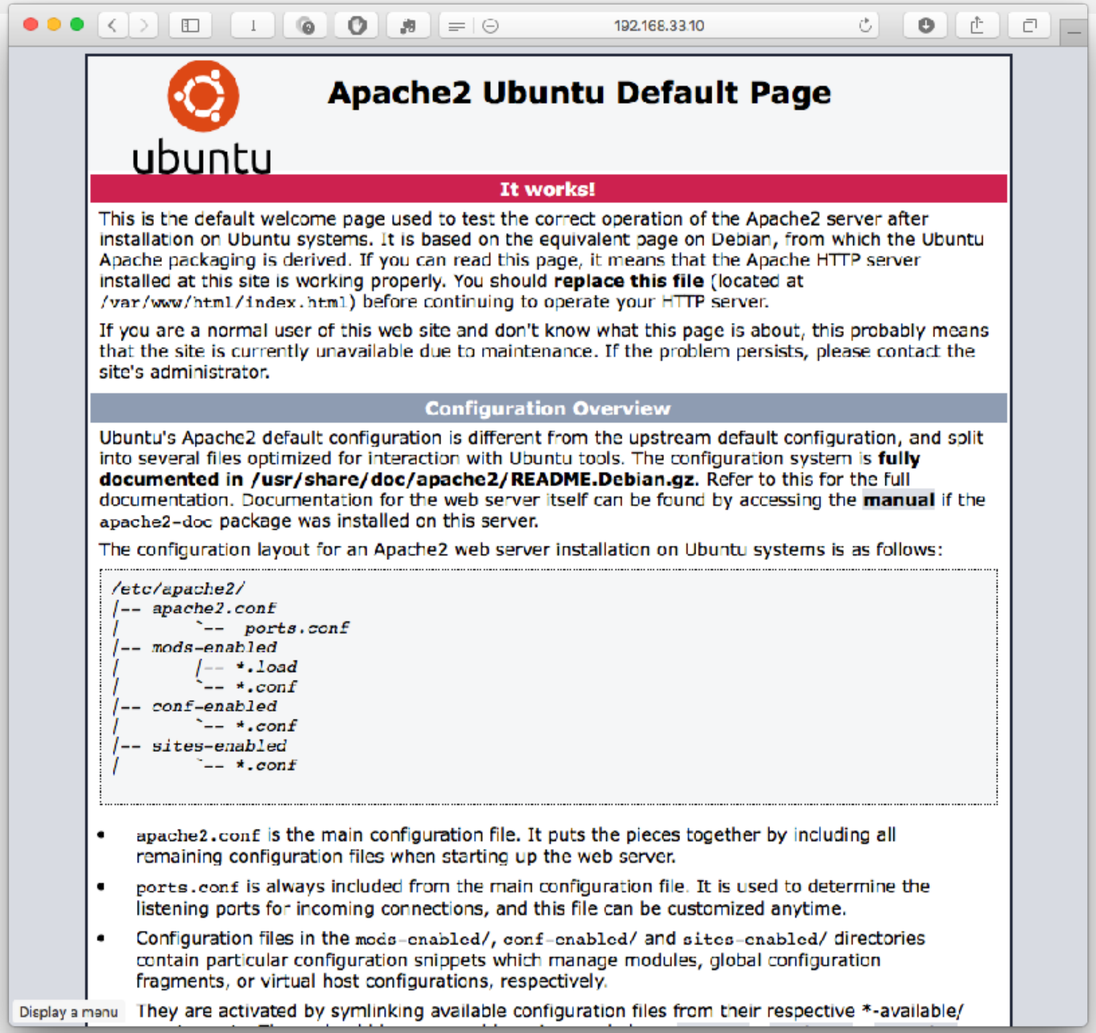
New Vagrant Run

```
iotia:lab-vagrant-demo rofrano$ vagrant reload
==> default: Attempting graceful shutdown of VM...
==> default: Checking if box 'ubuntu/bionic64' is up to date...
==> default: Clearing any previously set forwarded ports...
==> default: Couldn't find Cheffile at ./Cheffile.
==> default: Clearing any previously set network interfaces...
==> default: Preparing network interfaces based on configuration...
    default: Adapter 1: nat
    default: Adapter 2: hostonly
==> default: Forwarding ports...
    default: 80 (guest) => 8080 (host) (adapter 1)
    default: 22 (guest) => 2222 (host) (adapter 1)
==> default: Booting VM...
==> default: Waiting for machine to boot. This may take a few minutes...
    default: SSH address: 127.0.0.1:2222
    default: SSH username: ubuntu
    default: SSH auth method: password
    default: Warning: Remote connection disconnect. Retrying...
    default: Warning: Remote connection disconnect. Retrying...
==> default: Machine booted and ready!
GuestAdditions 5.0.20 running --- OK.
==> default: Checking for guest additions in VM...
==> default: Configuring and enabling network interfaces...
==> default: Mounting shared folders...
    default: /vagrant => /Users/rofrano/github/Demo/lab-vagrant-demo
==> default: Machine already provisioned. Run `vagrant provision` or use the `--provision`
==> default: flag to force provisioning. Provisioners marked to run always will still run.
iotia:lab-vagrant-demo rofrano$
```

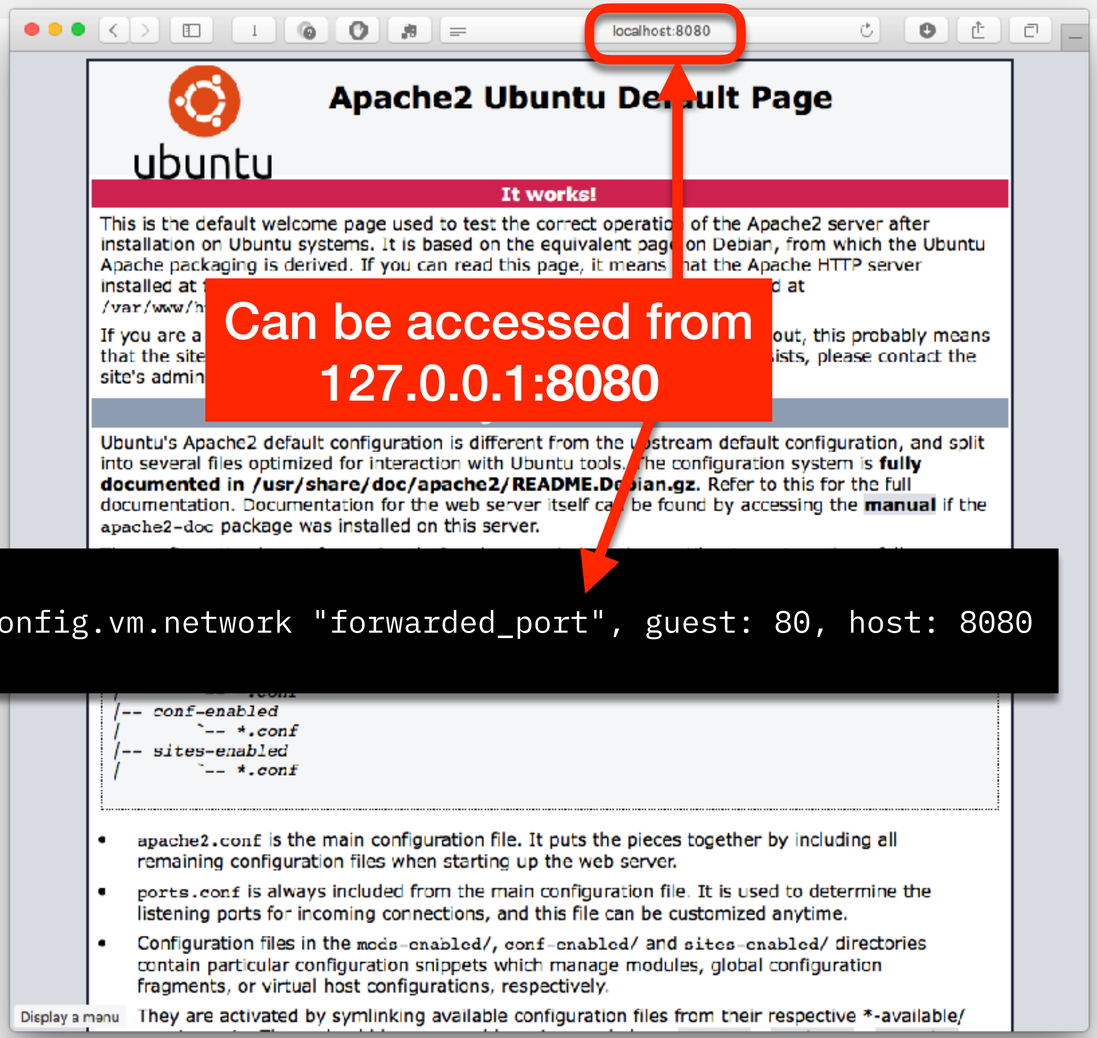
Our port got forwarded from 80 inside the VM to 8080 outside the VM

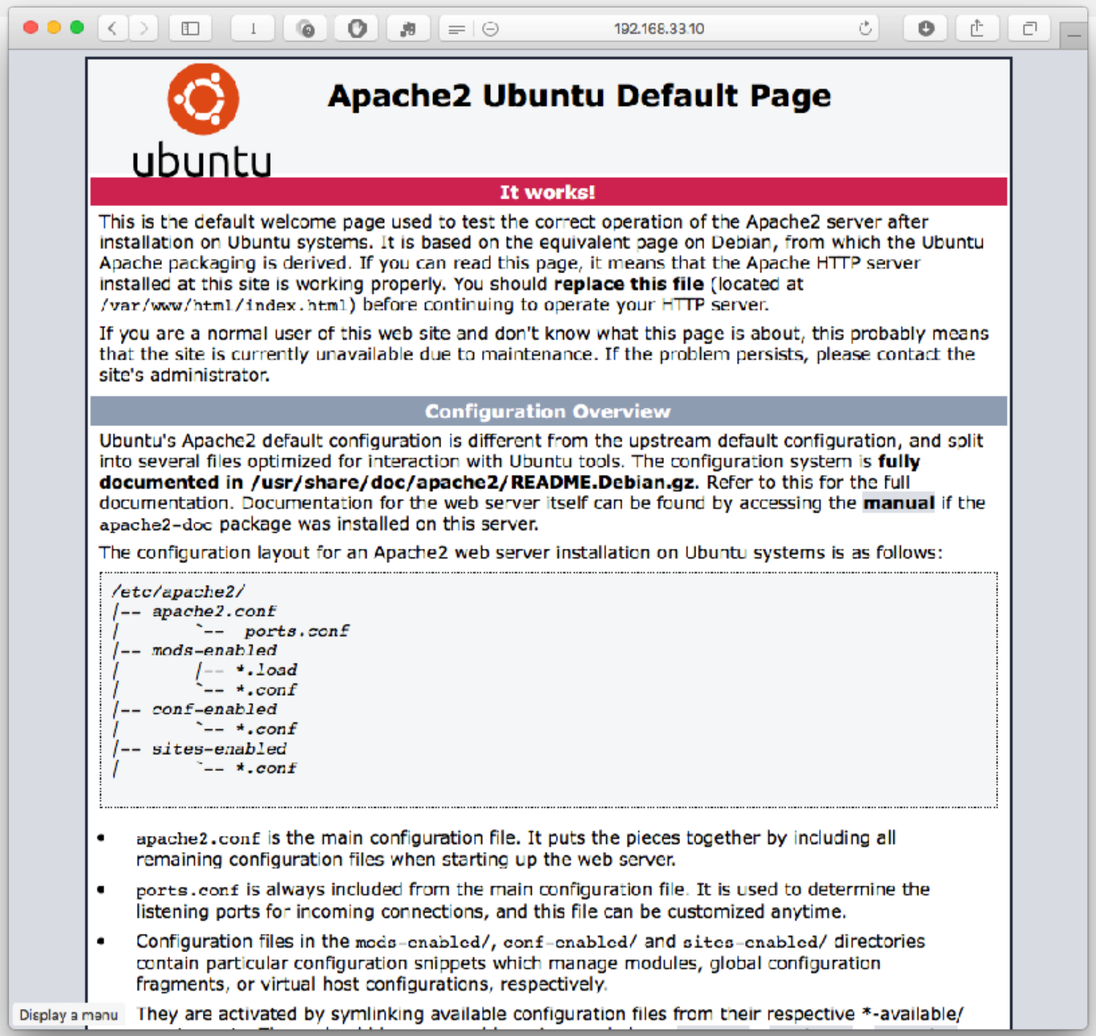
Test It In Your Browser



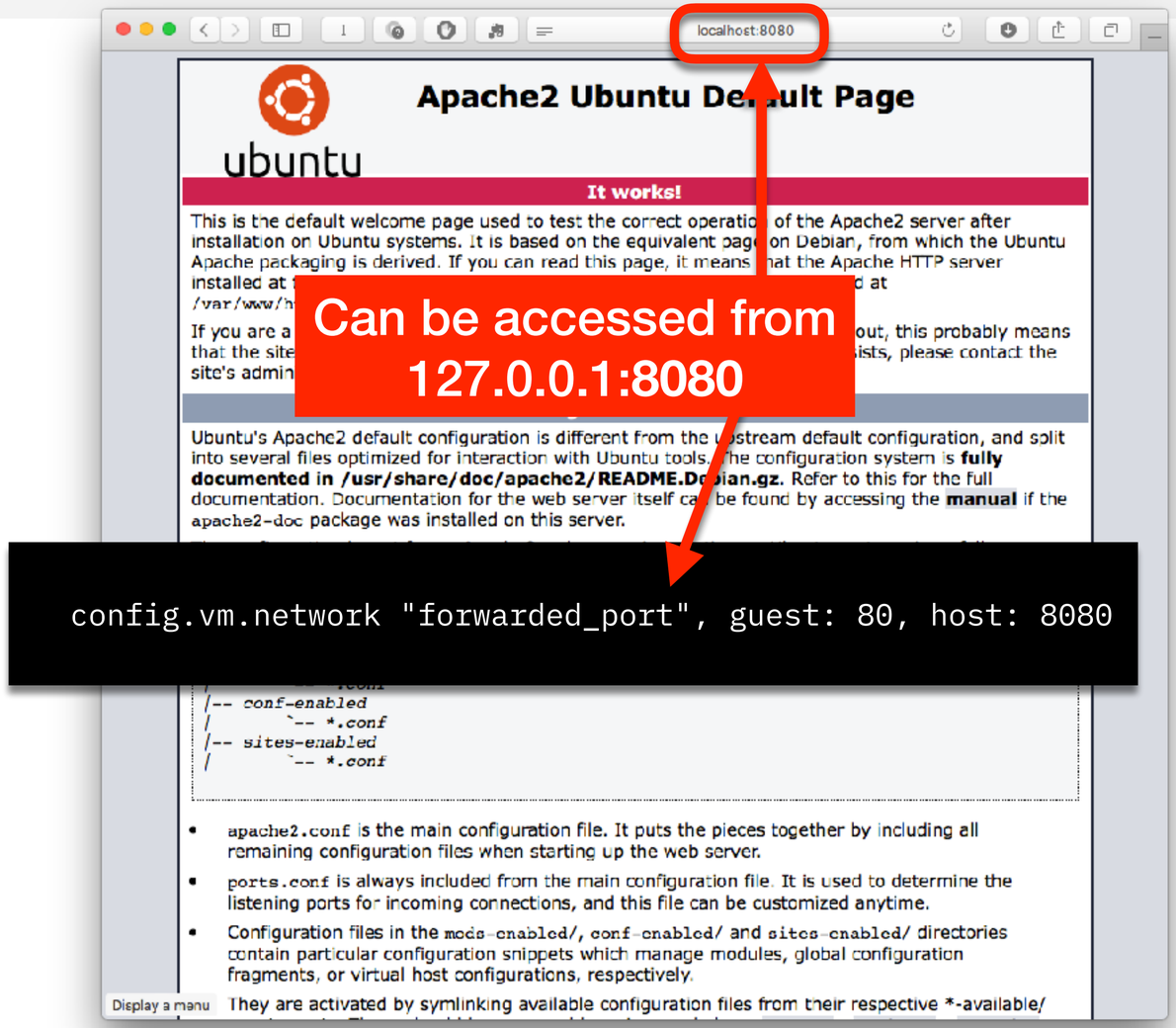


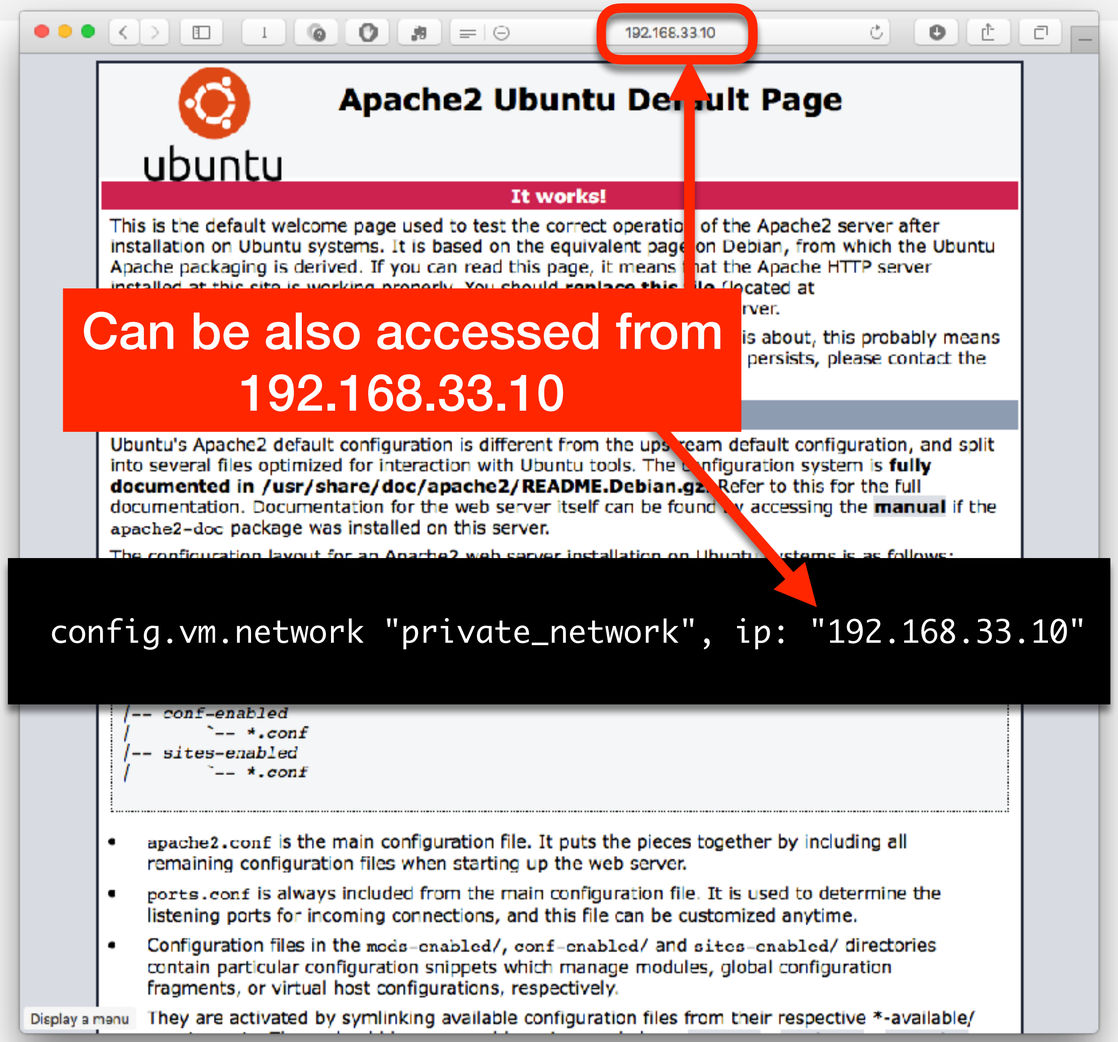
Test It In Your Browser





Test It In Your Browser





More Port Forwarding Examples

```
# Forward RAILS port
config.vm.network :forwarded_port, guest: 3000, host: 3000, auto_correct: true

# Forward Python Flask port
config.vm.network :forwarded_port, guest: 5000, host: 5000, auto_correct: true

# Forward PostgreSQL port
config.vm.network :forwarded_port, guest: 5432, host: 5432, auto_correct: true

# Forward MySQL port
config.vm.network :forwarded_port, guest: 3306, host: 3306, auto_correct: true

# Forward Redis port
config.vm.network :forwarded_port, guest: 6379, host: 6379, auto_correct: true

# Forward RabbitMQ Ports
config.vm.network :forwarded_port, guest: 15672, host: 15672, auto_correct: true
config.vm.network :forwarded_port, guest: 4369,  host: 4369,  auto_correct: true
config.vm.network :forwarded_port, guest: 5672,  host: 5672,  auto_correct: true
```

Adding Your Own Content

- We can link a local folder to the html root to serve up our own content (remember our local folder is shared with the VM!)
- First lets create a new `./html` folder and add `index.html`:

```
$ mkdir html
$ cd html
$ echo 'Hello From Vagrant!' > index.html
```

Link Our HTML Folder

- We can point `/var/www/html` to our host folder `/vagrant/`

```
Vagrant.configure(2) do |config|

  config.vm.box = “ubuntu/xenial64"

  # accessing "localhost:8080" will access port 80 on the guest machine.
  config.vm.network "forwarded_port", guest: 80, host: 8080

  config.vm.network "private_network", ip: "192.168.33.10"

  config.vm.provision "shell", inline: <<-SHELL
    apt-get update
    apt-get install -y apache2
    rm -rf /var/www/html
    ln -s /vagrant/html /var/www/html
  SHELL

end
```

Link Our HTML Folder

- We can point `/var/www/html` to our host folder `/vagrant/`

```
Vagrant.configure(2) do |config|

  config.vm.box = “ubuntu/xenial64"

  # accessing "localhost:8080" will access port 80 on the guest machine.
  config.vm.network "forwarded_port", guest: 80, host: 8080

  config.vm.network "private_network", ip: "192.168.33.10"

  config.vm.provision "shell", inline: <<-SHELL
    apt-get update
    apt-get install -y apache2
    rm -rf /var/www/html
    ln -s /vagrant/html /var/www/html
  SHELL

end
```

Remove the /var/www/html folder and link to our host’s ./html folder

Add 'rm' and 'ln' Commands

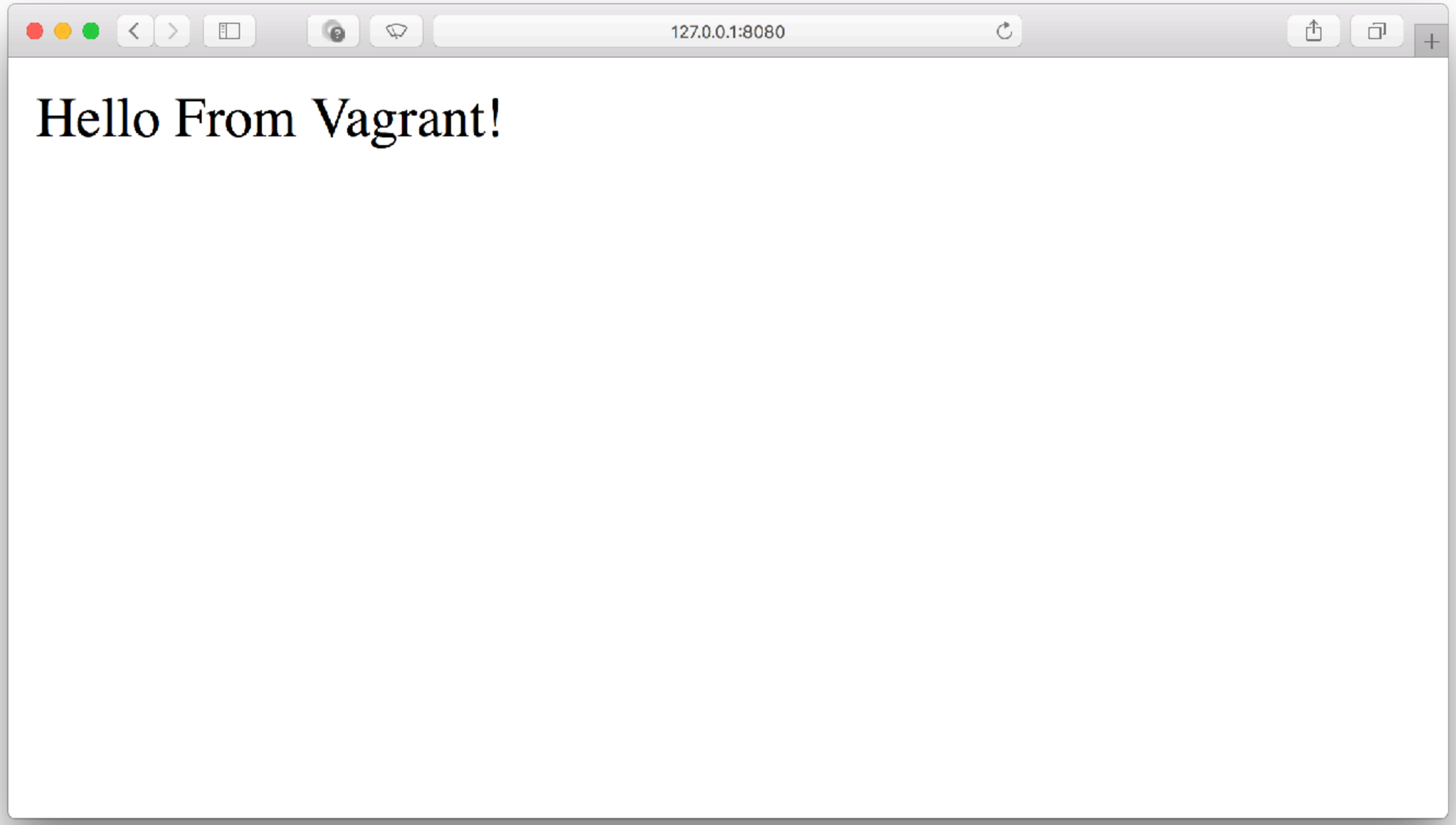
```
config.vm.provision "shell", inline: <<-SHELL
    apt-get update
    apt-get install -y apache2
    rm -rf /var/www/html
    ln -s /vagrant/html /var/www/html
SHELL
```

Re-provision the VM after Updates

- Since we changed a provision block in the `Vagrantfile` we must reload and provision

```
$ vagrant reload --provision
```

Apache is Serving Our Page



What Just Happened?

- We are serving up local files from our laptop inside of the VM
- That means we can develop locally with our OS IDE tools
- And we can run the code in a Linux VM just like a production server!
- Very Cool ! ! ! 😎

Summary

- You just deployed your first Vagrant environment
- You learned how to automatically install additional software
- You can now check your `Vagrantfile` into GitHub so that others can create the perfect development environment when working on your project.