

Vagrant Provisioners

IN A NUTSHELL

a talk by erikaheidi



whoami

- Brazilian, living in Amsterdam since 2012
- PHP developer for 10 years
- Working with independent projects
- Author of **Vagrant Cookbook** on LeanPub



Vagrant Provisioners in a Nutshell

What to expect from this talk

- 1) A quick guide on Vagrant
- 2) Provisioner Tasting: Ansible, Puppet and Chef
- 3) ProTips
- 4) Useful Resources



Revisiting **VAGRANT**

“It works on
my machine”

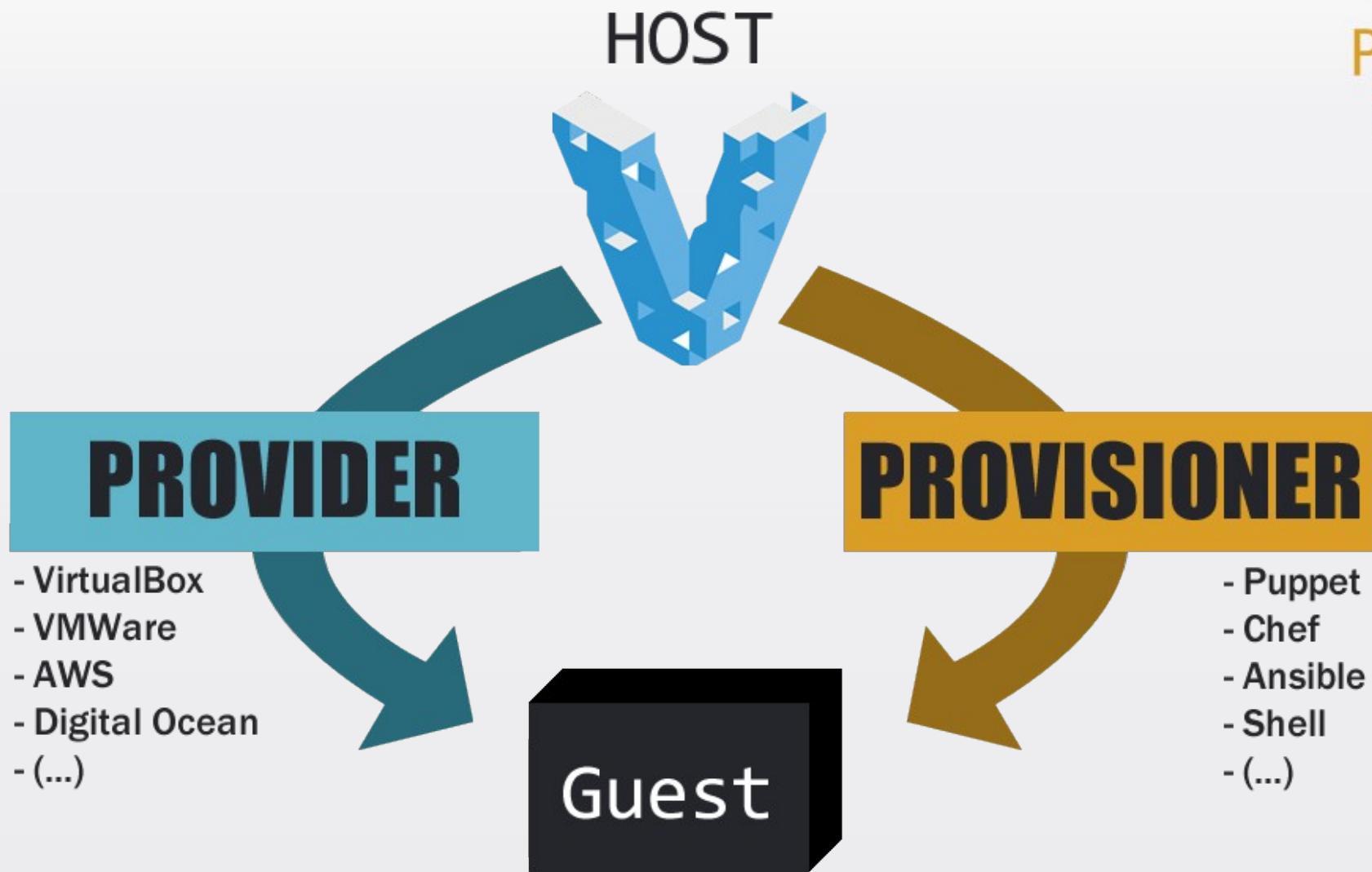
- every developer, ever

Why Vagrant?

- Reproducible and portable dev environment
- Enables easier code collaboration
- Backend env tests / benchmark
- Automation Tools learning and testing

What you need - basic

- Vagrant [1.4.x]
- VirtualBox [4.3.x]



The simplest thing that does something

```
Vagrant.configure("2") do |config|  
  
  config.vm.box = "precise64"  
  config.vm.box_url = "http://files.vagrantup.com/precise64.box"  
  
  config.vm.provision "shell",  
    inline: "echo hello, this is a simple Shell Provisioner!"  
  
end
```

```
erika@velvet: ~/Projects/vagrant-lab
→ vagrant-lab git:(master) ✘ vagrant up --provision
Bringing machine 'default' up with 'virtualbox' provider...
[default] Clearing any previously set forwarded ports...
[default] Clearing any previously set network interfaces...
[default] Preparing network interfaces based on configuration...
[default] Forwarding ports...
[default] -- 22 => 2222 (adapter 1)
[default] Running 'pre-boot' VM customizations...
[default] Booting VM...
[default] Waiting for machine to boot. This may take a few minutes...
[default] Machine booted and ready!
GuestAdditions 4.3.6 running --- OK.
[default] Configuring and enabling network interfaces...
[default] Mounting shared folders...
[default] -- /vagrant
[default] Running provisioner: shell...
[default] Running: inline script
stdin: is not a tty
hello, this is a simple Shell Provisioner!
→ vagrant-lab git:(master) ✘ █
```

Vagrant Provisioners in a Nutshell

VAGRANT CHEAT SHEET

boots the virtual machine and apply the configured provisioners

up

```
$ vagrant up
```

logs in using default settings (no password is needed)

ssh

```
$ vagrant ssh
```

reloads the box, booting the machine again.

reload

```
$ vagrant reload
```

runs only the provisioners, without reloading / rebooting vm

provision

```
$ vagrant provision
```

turns off the VM

halt

```
$ vagrant halt
```

suspends the VM
(saves the machine state)

suspend

```
$ vagrant suspend
```

resumes a previously suspended VM

resume

```
$ vagrant resume
```

destroys the VM - for when you want to start from scratch

destroy

```
$ vagrant destroy
```



Provisioner TASTING



1. Ansible



Syntax	YAML
Scripts	Playbooks
Execution Order	Sequential
Modularity	Many built-in modules
Popularity	Third most used provisioner
Documentation	Clear, objective

Note: requires installation of extra package (**ansible**).

1.1 Vagrantfile

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

1.2 Ansible: playbook

```
#simple playbook example
---
- hosts: all
  sudo: true
Tasks:
  - name: Update apt
    apt: update_cache=yes

  - name: Install Nginx
    apt: pkg=nginx state=latest
```

```
[default] Configuring and enabling network interfaces...
[default] Mounting shared folders...
[default] -- /vagrant
[default] Running provisioner: ansible...

PLAY [all] ****
GATHERING FACTS ****
ok: [default]

TASK: [Update apt] ****
ok: [default]

TASK: [Install Nginx] ****
changed: [default]

PLAY RECAP ****
default : ok=3    changed=1    unreachable=0    failed=0
```

Vagrant Provisioners in a Nutshell

2. Puppet (puppet-apply)



Syntax	Custom based on Ruby
Scripts	Manifests
Execution Order	NOT Sequential
Modularity	Very easy to find modules on the Internet
Popularity	Most used provisioner for Vagrant
Documentation	A bit confusing

2.1 Vagrantfile

```
config.vm.provision :puppet do |puppet|
    puppet.module_path = "modules"
end
```

2.2 Puppet: manifest

```
#manifests/default.pp

Exec { path => [ "/bin/" , "/sbin/" , "/usr/bin/" , "/usr/sbin/" ] }

exec { 'apt-get update':
  command => 'apt-get update',
}

package { 'nginx':
  ensure => "installed",
  require => Exec['apt-get update'],
}
```

```
[default] Configuring and enabling network interfaces...
[default] Mounting shared folders...
[default] -- /vagrant
[default] -- /tmp/vagrant-puppet-1/manifests
[default] Running provisioner: puppet...
Running Puppet with default.pp...
stdin: is not a tty
warning: Could not retrieve fact fqdn
info: Applying configuration version '1389029896'
notice: /Stage[main]//Exec[apt-get update]/returns: executed successfully
notice: /Stage[main]//Package[nginx]/ensure: ensure changed 'purged' to 'present'
info: Creating state file /var/lib/puppet/state/state.yaml
notice: Finished catalog run in 12.48 seconds
→ puppet git:(master) ✘
```

3. Chef (chef_solo)



Syntax	Ruby
Scripts	Recipes
Execution Order	Sequential
Modularity	Many “cookbooks” available on the Internet
Popularity	Second most used provisioner
Documentation	Chaos!

3.1 Vagrantfile

```
config.vm.provision "chef_solo" do |chef|
    chef.add_recipe "nginx"
end
```

3.2 Chef: recipe

```
#cookbooks/nginx/recipes/default.rb
execute "apt-get update" do
  command "apt-get update"
end
apt_package "nginx" do
  action :install
end
```

```
[default] Configuring and enabling network interfaces...
[default] Mounting shared folders...
[default] -- /vagrant
[default] -- /tmp/vagrant-chef-1/chef-solo-1/cookbooks
[default] Running provisioner: chef_solo...
Generating chef JSON and uploading...
Running chef-solo...
stdin: is not a tty
[2014-01-06T18:05:09+00:00] INFO: *** Chef 10.14.2 ***
[2014-01-06T18:05:09+00:00] INFO: Setting the run_list to ["recipe[nginx]"] from JSON
[2014-01-06T18:05:09+00:00] INFO: Run List is [recipe[nginx]]
[2014-01-06T18:05:09+00:00] INFO: Run List expands to [nginx]
[2014-01-06T18:05:09+00:00] INFO: Starting Chef Run for precise64
[2014-01-06T18:05:09+00:00] INFO: Running start handlers
[2014-01-06T18:05:09+00:00] INFO: Start handlers complete.
[2014-01-06T18:05:17+00:00] INFO: execute[apt-get update] ran successfully
[2014-01-06T18:05:23+00:00] INFO: Chef Run complete in 13.615525 seconds
[2014-01-06T18:05:23+00:00] INFO: Running report handlers
[2014-01-06T18:05:23+00:00] INFO: Report handlers complete
→ chef git:(master) ✘
```

Vagrant Provisioners in a Nutshell



PRO
TIPS

6.1 Debugging

- Unknown Vagrant error
 - Use VirtualBox / Vmware GUI
- Unknown Provisioner error
 - Increase provisioner verbosity
- Not working as expected
 - Login, fix, automate

6.2 NFS Performance

- Synchronization has a cost
- Symfony cache/logs
 - Too much writing operations on disk
 - We don't need this in our synced folder

A screenshot of a local server setup, likely using Vagrant, showing a web application running on a local host. The browser window displays a landing page for "FotoMontagens" with various photo editing tools and features. A prominent orange arrow points from the top right towards the browser window, highlighting the application. Another orange arrow points from the bottom right towards a performance metric displayed on the screen.

The browser tabs show the application is running on `fm.local/app_dev.php`. The main content area shows a sidebar with the "local server" logo and several menu items: MOLDURAS, EFEITOS, FOTO ANIMADA, REVISTAS, MAIS, and AJUDA. A central modal window is open, titled "9 beleggingsfouten die u in 2014 moet voorkomen", containing text about investment mistakes and a download link. Below this, a section titled "MONTAGENS EM DESTAQUE" shows four thumbnail images labeled "Test Vagrant", "Test Vagrant 2", "Test Vagrant 3", and "Test Vagrant 4". Each thumbnail has a small edit icon and a placeholder text below it: "test, dummy, placephant". To the right of these thumbnails, a large orange lightning bolt graphic contains the text "206 ms" next to a stopwatch icon, indicating the loading time of the page.

At the bottom of the browser window, the developer toolbar shows various status icons and metrics: "2.3.7", "php", "677a85 app | dev", "200", "DefaultController indexAction on fotomontagens_site_homepage", "206 ms", "18.5 MB", "anon.", and a session ID "1".

Vagrant Provisioners in a Nutshell

A screenshot of a Google Chrome browser window displaying a web application at 192.168.33.101/app_dev.php. The application interface includes a top navigation bar with links like 'MOLDURAS', 'EFEITOS', 'FOTO ANIMADA', 'REVISTAS', 'MAIS', and 'AJUDA'. A sidebar on the left features a logo for 'vm server' and several thumbnail previews of photo montages. A large orange arrow points from the right side of the screen towards the application's interface. In the bottom right corner of the application, there is a performance metric box showing '6899 ms' with a 'L' icon.

FotoMontagens - Montagens de Fotos Online Grátis - Google Chrome

Google Translate | Inbox - erika@er... | Entrada (123) | Recent - Google | MesaBjörn [Fo... | Google Translate | online charts - Go... | csvkit 0.6.2 (beta) | csvgrep — csvkit | infographic - Goo... | Criar Montagens | FotoMontagens -

192.168.33.101/app_dev.php

Development Services TO READ Fitness Amsterdam Inspiration, Ideas Templates Soft Skills New folder GeekFun DutchNews.nl b...

3:11 PM erika

Crie montagens e personalize suas fotos gratuitamente no fotomontagens.net!

vm server

MOLDURAS EFEITOS FOTO ANIMADA REVISTAS MAIS AJUDA

Crie molduras prontas para personalizar suas fotos Aplique efeitos e filtros em suas fotos Crie gifs animados com suas fotos Crie capas de revista com suas fotos Montagens Especiais e outras ferramentas Perguntas Fre Tutores

ofm

24.2

MONTAGENS EM DESTAQUE

Test Vagrant Test Vagrant 2 Test Vagrant 3 Test Vagrant 4

test, dummy, placephant

test, dummy, placephant

test, dummy, placephant

test, dummy, placephant

FRESH & CLEAN DESIGN
Quisque ut nisl et neque blandit quis interdum lorem massa varius.

FULLY RESPONSIVE
Quisque ut nisl et neque blandit quis interdum lorem massa varius.

VERY FLEXIBLE
Quisque ut nisl et neque blandit quis interdum lorem massa varius.

ONGOING SUPPORT
Quisque ut nisl et neque blandit quis interdum lorem massa varius.

6899 ms

Vagrant Provisioners in a Nutshell

A screenshot of a Google Chrome browser window displaying a web application. The application's logo, 'ofm', is visible in the top left. The main content area features a large orange banner with the text 'optimized vm server'. Below this, there's a section titled 'MONTAGENS EM DESTAQUE' (Featured Montages) showing four thumbnail images labeled 'Test Vagrant', 'Test Vagrant 2', 'Test Vagrant 3', and 'Test Vagrant 4'. Each thumbnail has a small edit icon and the placeholder text 'test, dummy, placephant'. Below these thumbnails are four descriptive cards: 'FRESH & CLEAN DESIGN' (star icon), 'FULLY RESPONSIVE' (phone icon), 'VERY FLEXIBLE' (gear icon), and 'ONGOING SUPPORT' (cog icon). A large orange arrow points diagonally across the page from the bottom left towards the top right, highlighting the application's features. The browser's address bar shows the URL '192.168.33.101/app_dev.php'. The status bar at the bottom shows various system icons and the text 'DefaultController indexAction on fotomontagens_site_homepage 111 ms 6.0 MB anon. 1'.

Vagrant Provisioners in a Nutshell



MORE RESOURCES

Vagrant Provisioners in a Nutshell



PuPHPet

A simple GUI to set up virtual machines for Web development.



Free Web App
Performance Monitoring
and Shirt When You Try
New Relic

ads via Carbon

Deploy Target

- Local
- Digital Ocean
- Rackspace
- Amazon EC2

Server Basics

- MailCatcher

Webserver

- Apache
- Nginx

PHP Engine

- Official PHP
- Facebook HHVM

Database

- MySQL
- PostgreSQL

Have an existing PuPHPet-generated manifest? Just drag your `puphpet/config.yaml` file into your browser and the form will be filled in with your previous values! ×

Deploy Target

[Local](#) [Digital Ocean](#) [Rackspace](#) [Amazon Web Services](#)

Local VM Instructions

Pre-requisites

1. Download the latest version of VirtualBox from [here](#)
2. Download the latest version of Vagrant from [here](#).

Vagrant Provisioners in a Nutshell

PHP dev environments powered by Vagrant and Ansible

Phansible provides a simple interface for generating a basic Vagrant provision for PHP development environments, using Ansible. Think about it as a Vagrant bootstrap generator.

Bundle Generator

VM Settings

VM Name	default	IP (private network)	192.168.33.99
Base Box	Ubuntu Precise Pangolin 64	Shared Folder	/
Memory (MB)	512		

Web Server

- NGINX + PHP5-FPM
- APACHE + PHP5
- NGINX + HHVM
- Install Composer

Document Root:	/vagrant
PHP Version:	<input type="radio"/> 5.4 <input checked="" type="radio"/> 5.5

PHP Packages

- php5-cli
- php-pear
- php5-curl
- php5-imagick

Other Packages

Vagrant Provisioners in a Nutshell

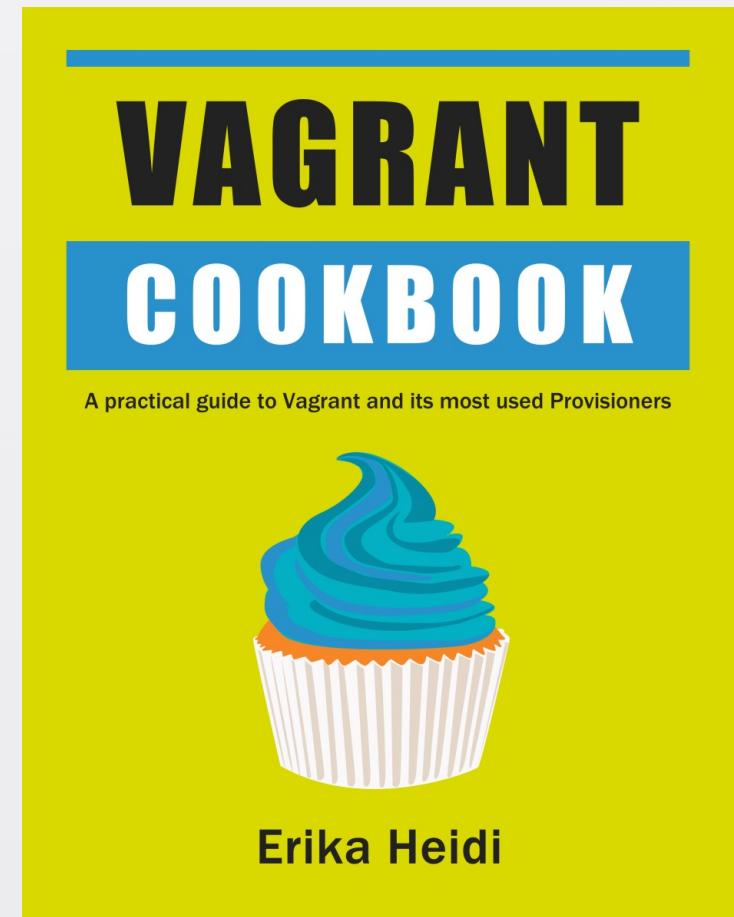
Quickly getting started

- SandBox PHP
 - Ansible, Puppet and Chef
 - <https://github.com/vagrantee/sandbox-php>



Vagrant Cookbook

Special discount coupon for PHP UK:



<http://leanpub.com/vagrantcookbook/c/phpuk14>

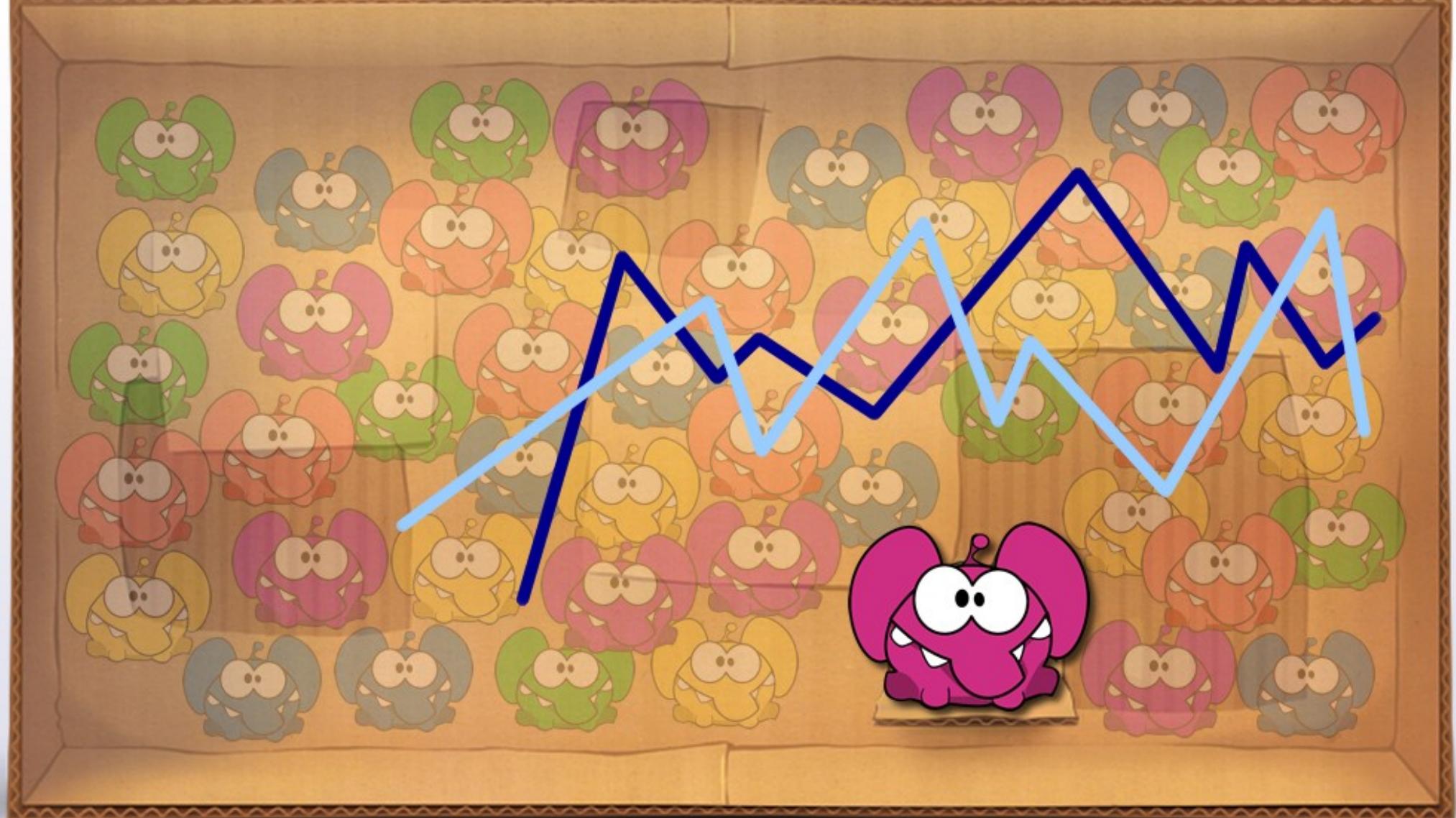
Vagrant Provisioners in a Nutshell

QUESTIONS



THANKS!!!

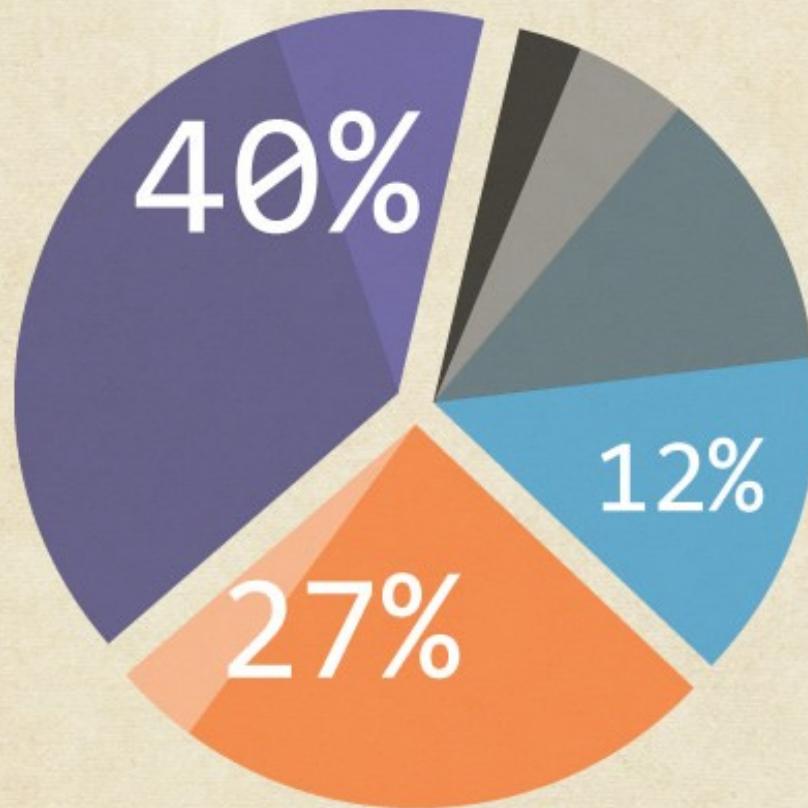
erikaheidi.com/vagrant
joind.in/10712



Vagrant usage research



VAGRANT



PUPPET

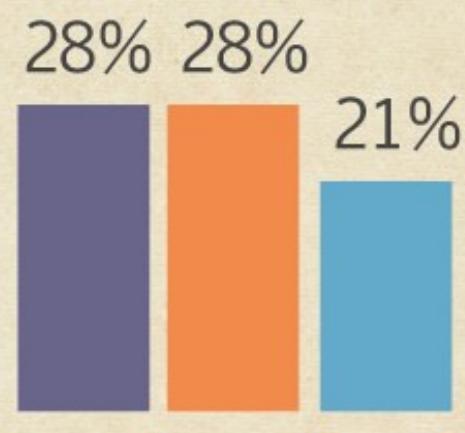
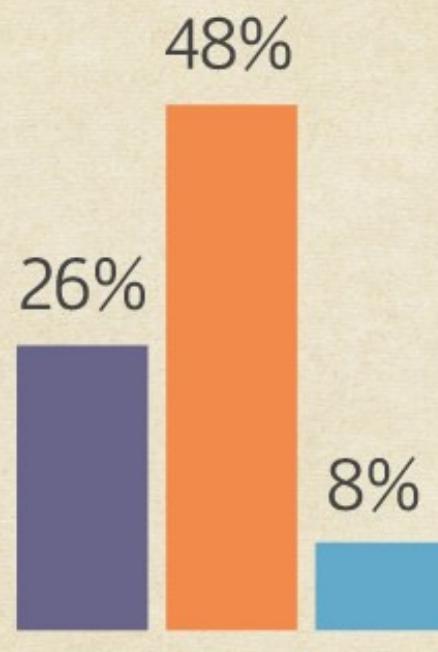
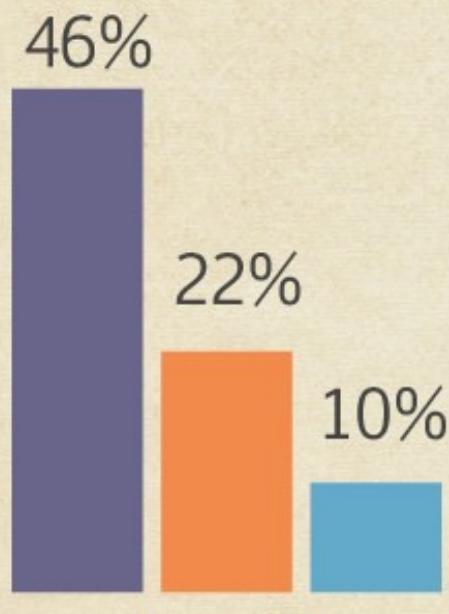
is the most popular
with **40%**
(apply+agent)

puppet apply	31%
puppet agent	9%
chef solo	23%
chef client	4%
ansible	12%
salt	3%
shell	6%
others	12%



VAGRANT

...except with Ruby developers.



Puppet

Chef

Ansible



USAGE RESEARCH

JANUARY 2014

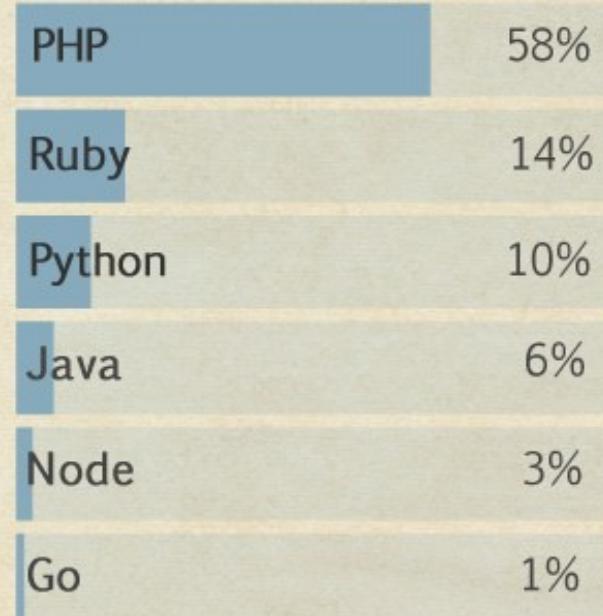
BOXES

58%

of boxes
run apps
written in



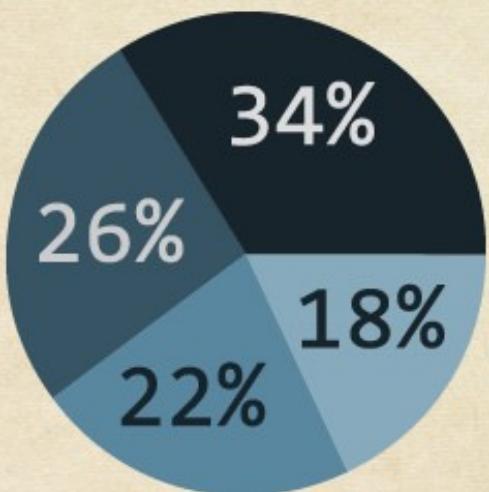
PHP



65% are OSX Users



■ OSX Users ■ Linux Users ■ Windows Users



66% of participants
use Vagrant for
less than 1 year

- less than 3 months
- 3 to 6 months
- 6 months to 1 year
- more than 1 year

Why Vagrant?

To develop and test applications	616
To share the same environment between coworkers	460
To avoid messing with my Host OS	421