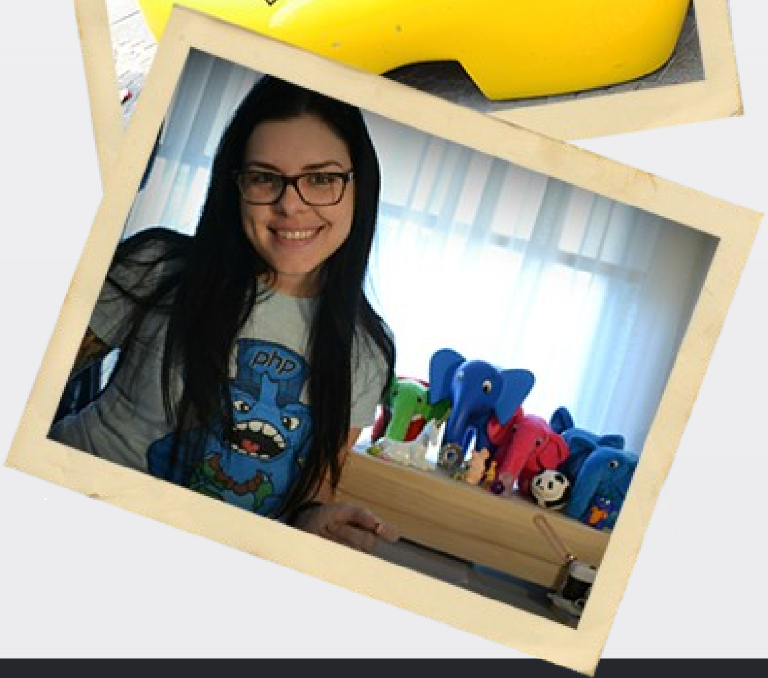
# Vagrant Provisioners

## 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



PHPUK
2014

## 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?

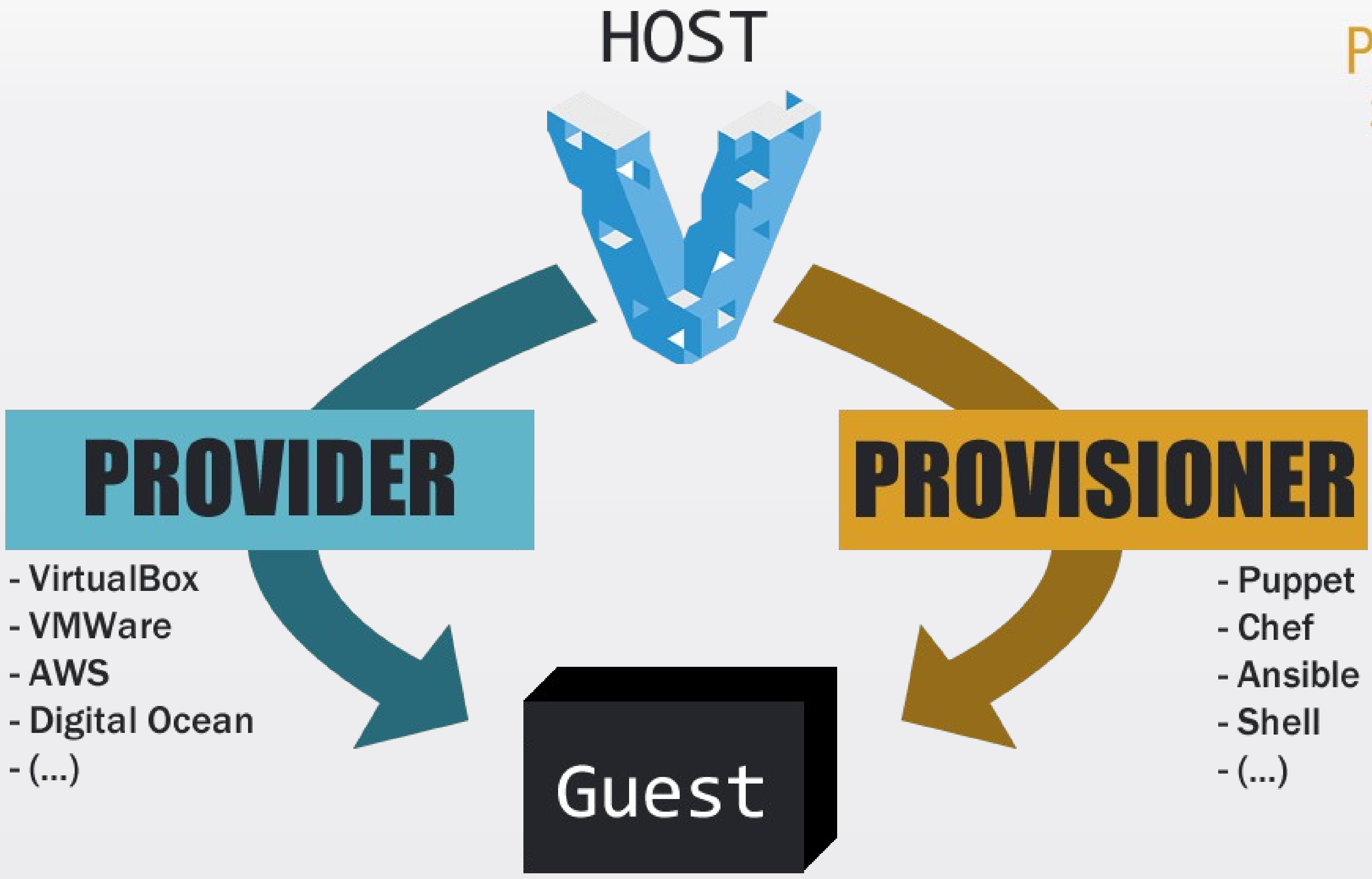
PHPUK 2014

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

PHPUK 2014

## What you need - basic

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



- VirtualBox
- VMWare
- AWS
- Digital Ocean
- (...)

- Puppet
- Chef
- Ansible
- Shell
- (...)

## Vagrant Provisioners in a Nutshell

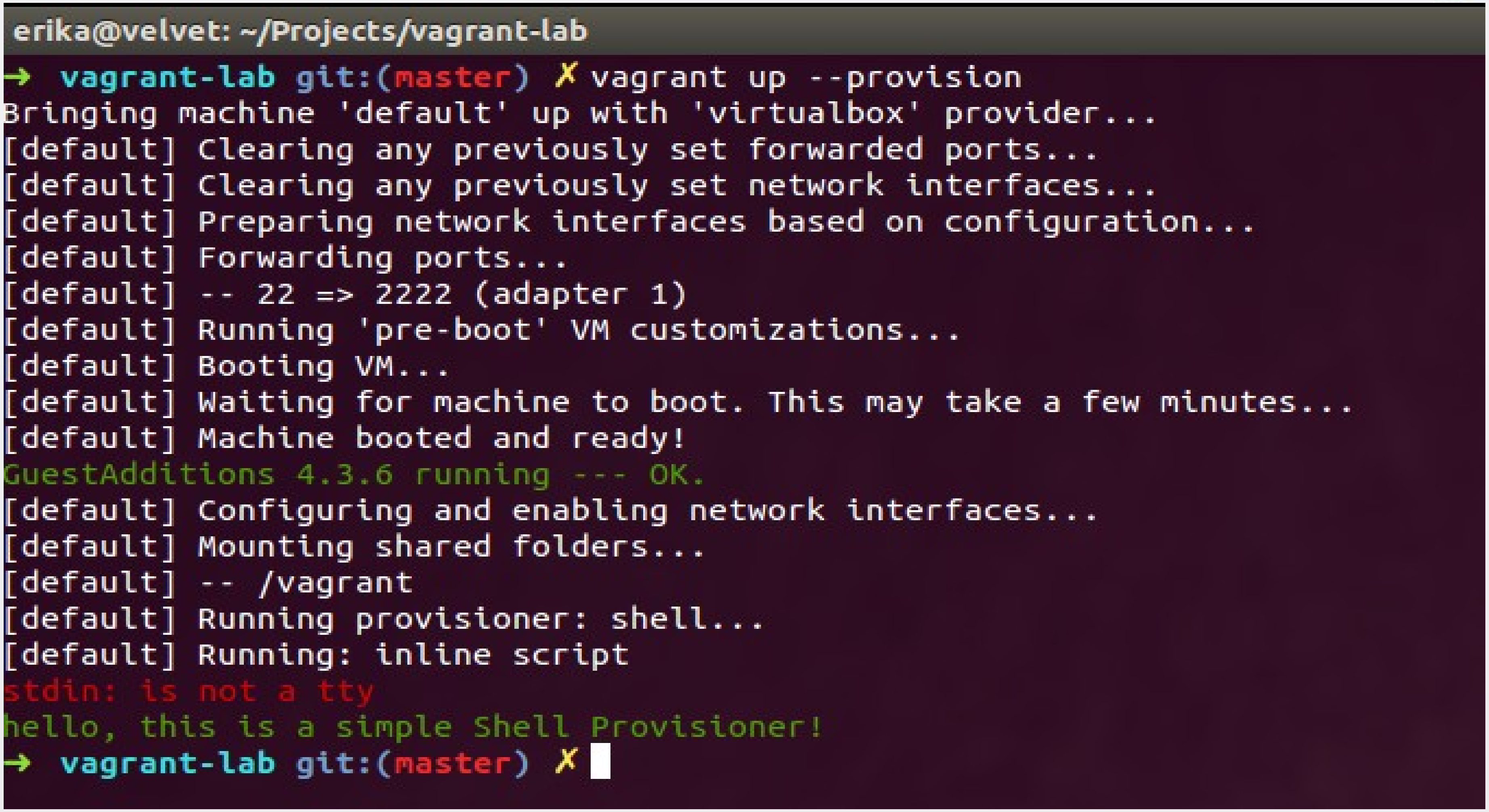
## The simplest thing that does something

```
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) ✗
```

PHPUK 2014

## 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

ANSIBLE WORKS puppet labs Chef

## 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

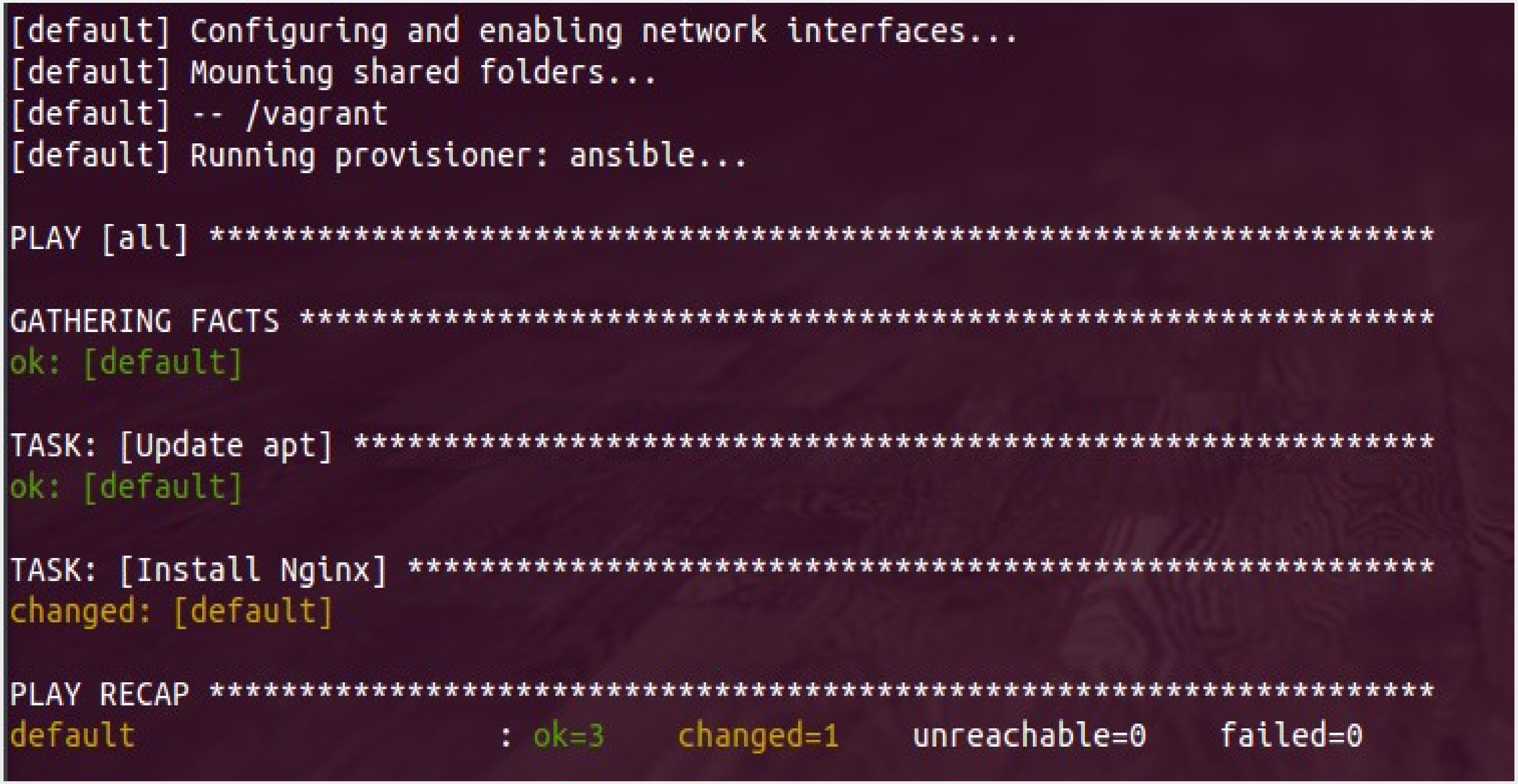
PHPUK 2014

```
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
```

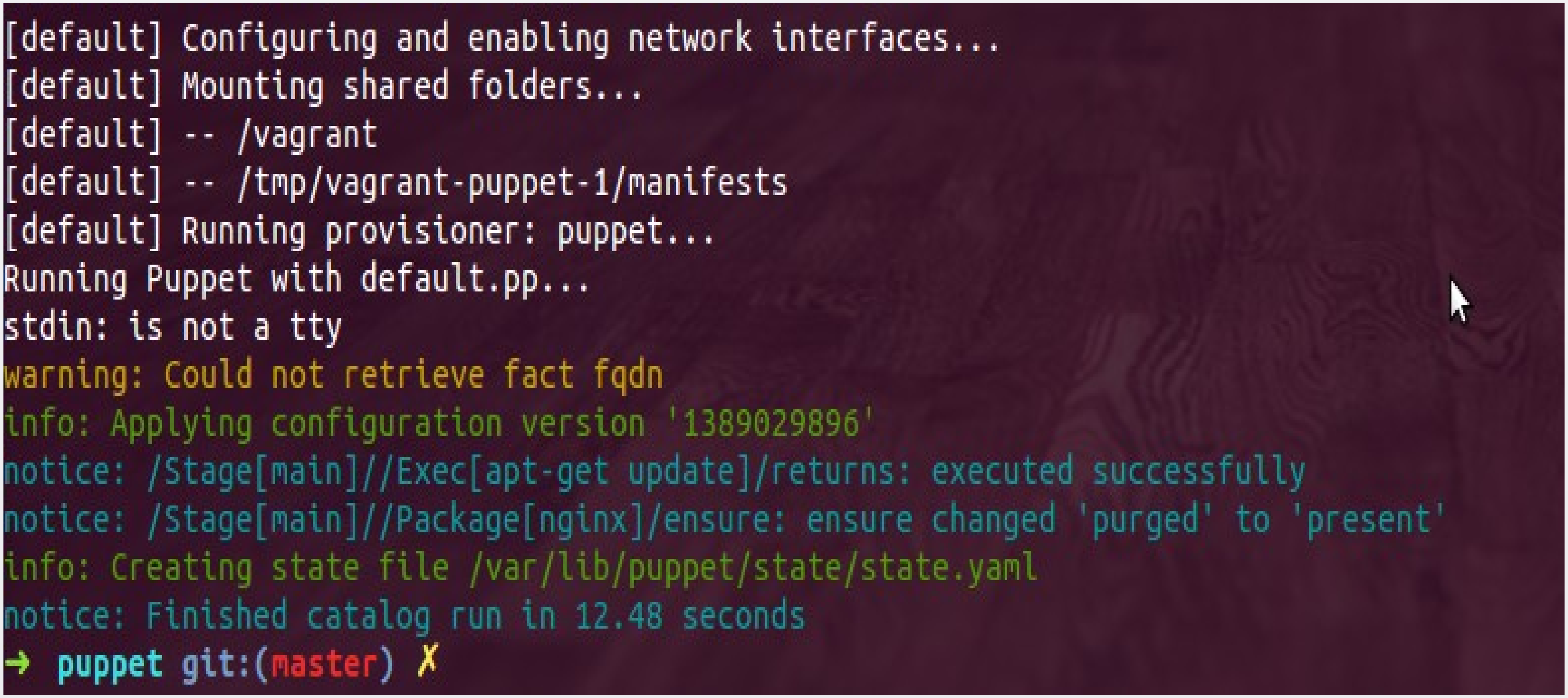
## 2.2 Puppet: manifest

```
#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) ✗
```

Vagrant Provisioners in a Nutshell

## 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! |

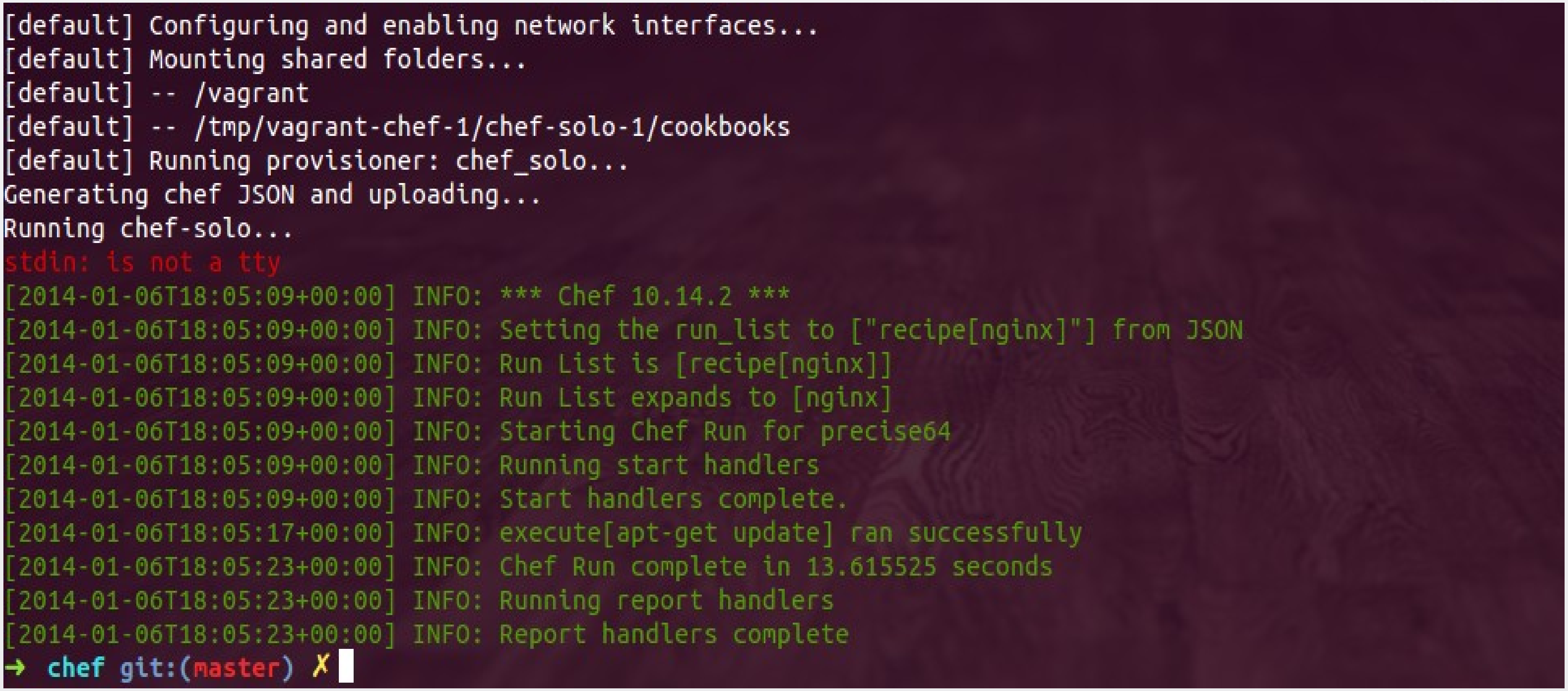
## 3.1 Vagrantfile

```
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
```

PHPUK
2014



```
[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

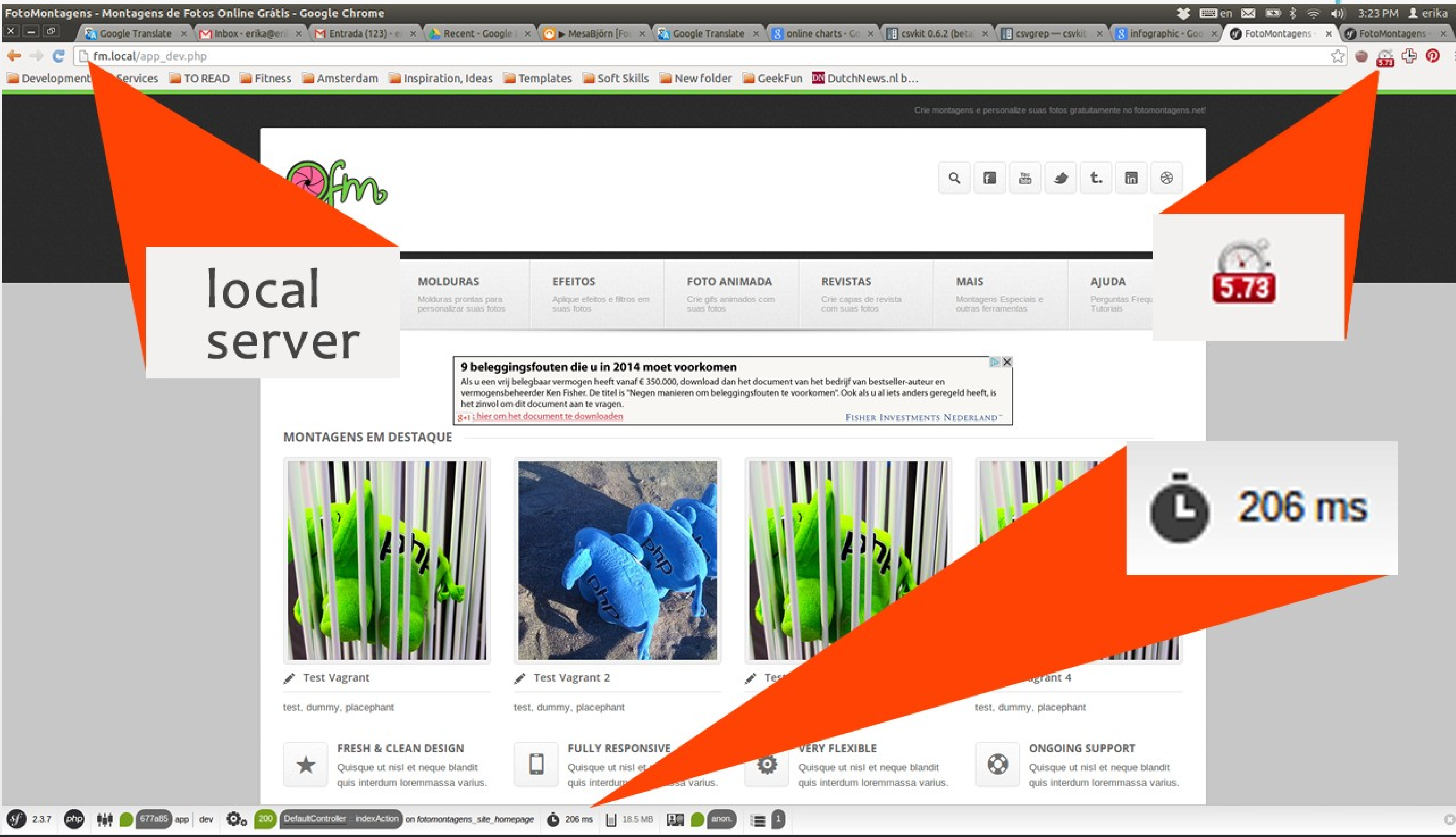
## 6.1 Debugging

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

PHPUK
2014

## 6.2 NFS Performance

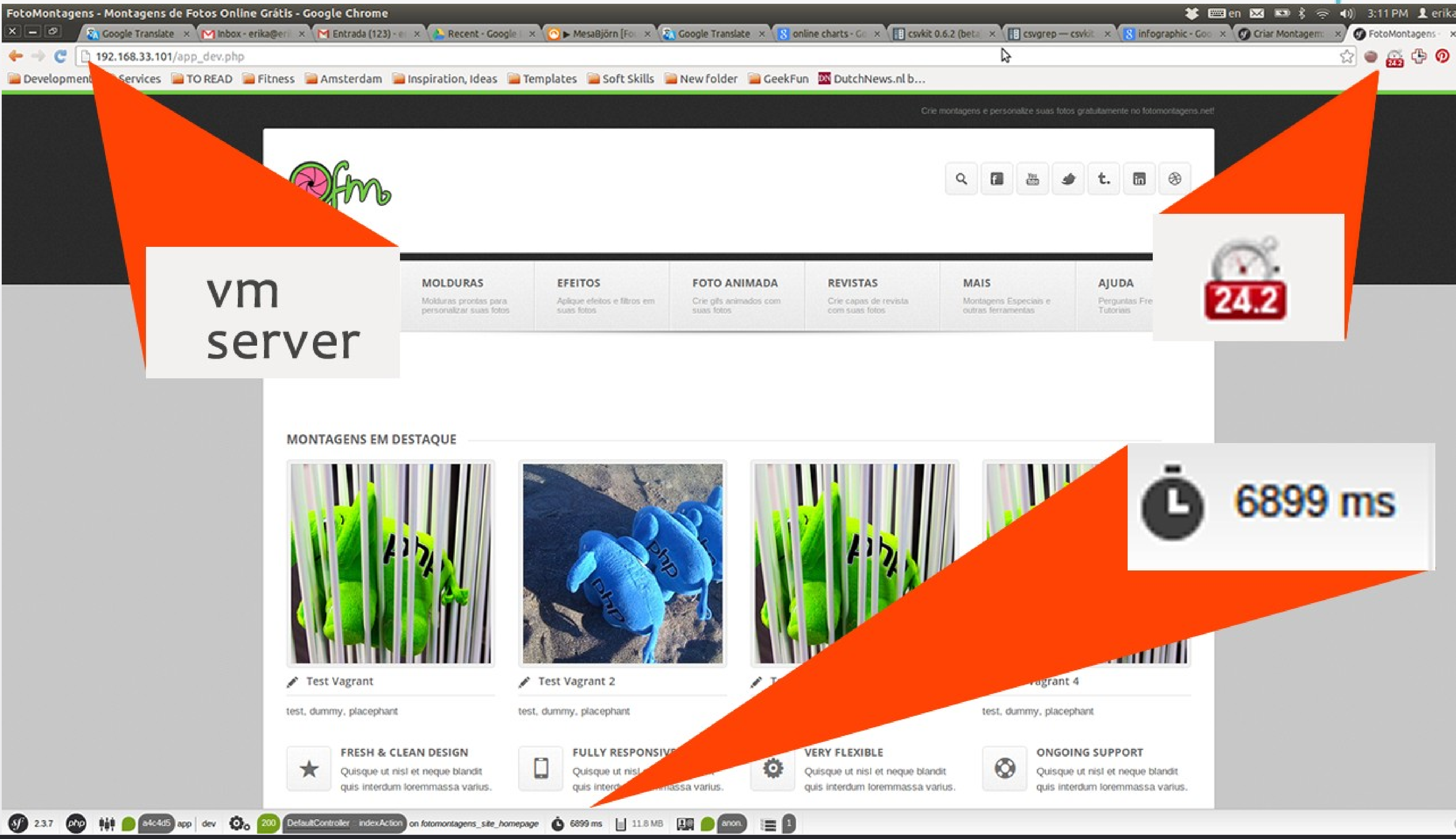
- Synchronization has a cost
- Symfony cache/logs
  - Too much writing operations on disk
  - We don't need this in our synced folder



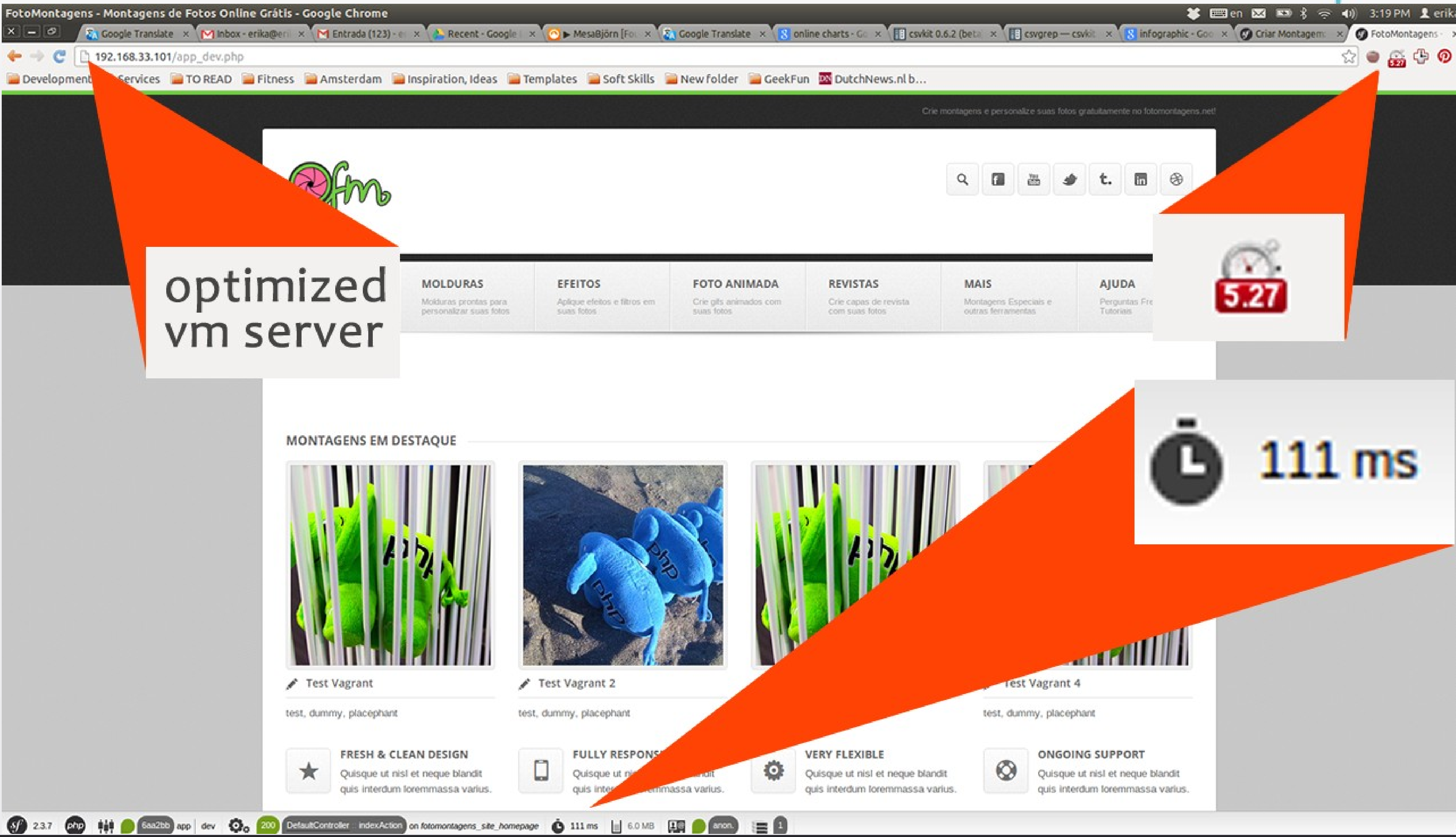
9 beleggingsfouten die u in 2014 moet voorkomen

Als u een vrij belegbaar vermogen heeft vanaf € 350.000, download dan het document van het bedrijf van bestseller-auteur en vermogensbeheerder Ken Fisher. De titel is "Negen manieren om beleggingsfouten te voorkomen". Ook als u al iets anders geregeld heeft, is het zinvol om dit document aan te vragen.

## Vagrant Provisioners in a Nutshell



## Vagrant Provisioners in a Nutshell

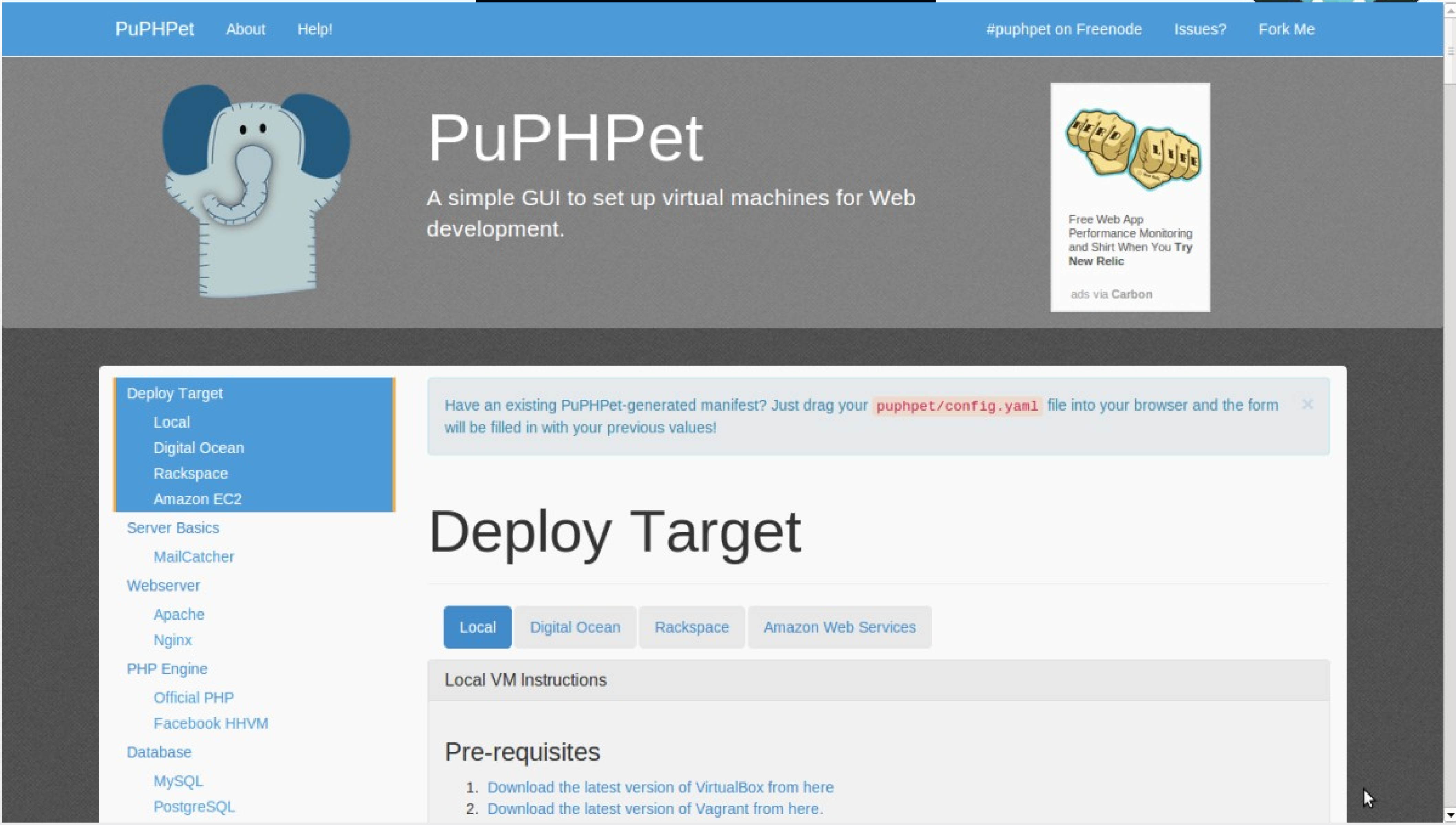


## Vagrant Provisioners in a Nutshell

PHPUK
2014

## MORE RESOURCES

# puphpet.com



PuPHPet About Help! #puphpet on Freenode Issues? Fork Me

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

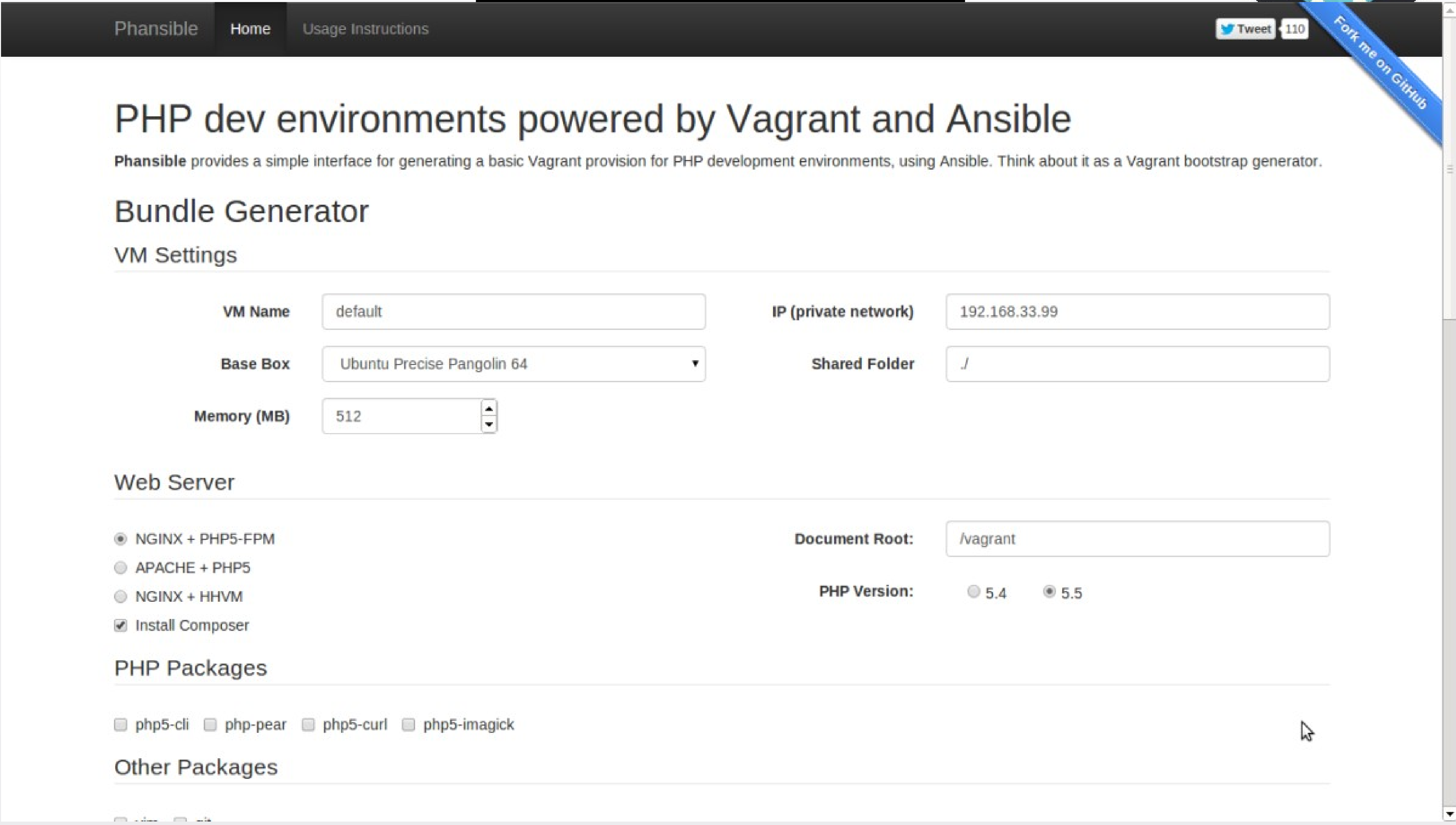
Local VM Instructions

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

# phansible.com



## 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:** 5.4 [✓ 5.5]

#### PHP Packages

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

#### Other Packages

## Vagrant Provisioners in a Nutshell

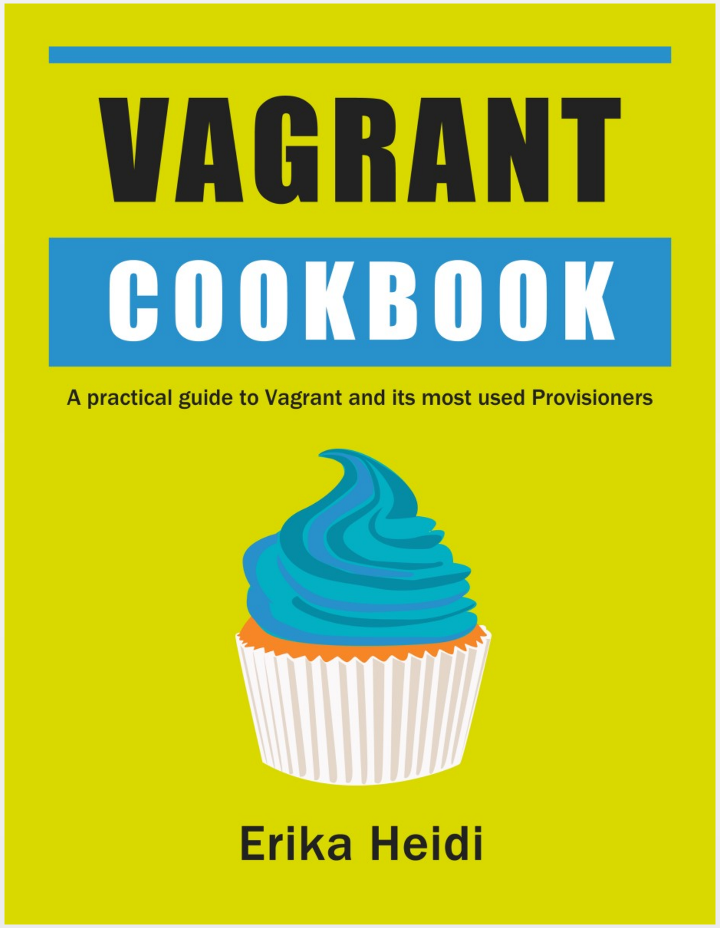
## Quickly getting started

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

## Vagrant Cookbook

Special discount coupon for PHPUK:

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



VAGRANT

COOKBOOK

A practical guide to Vagrant and its most used Provisioners

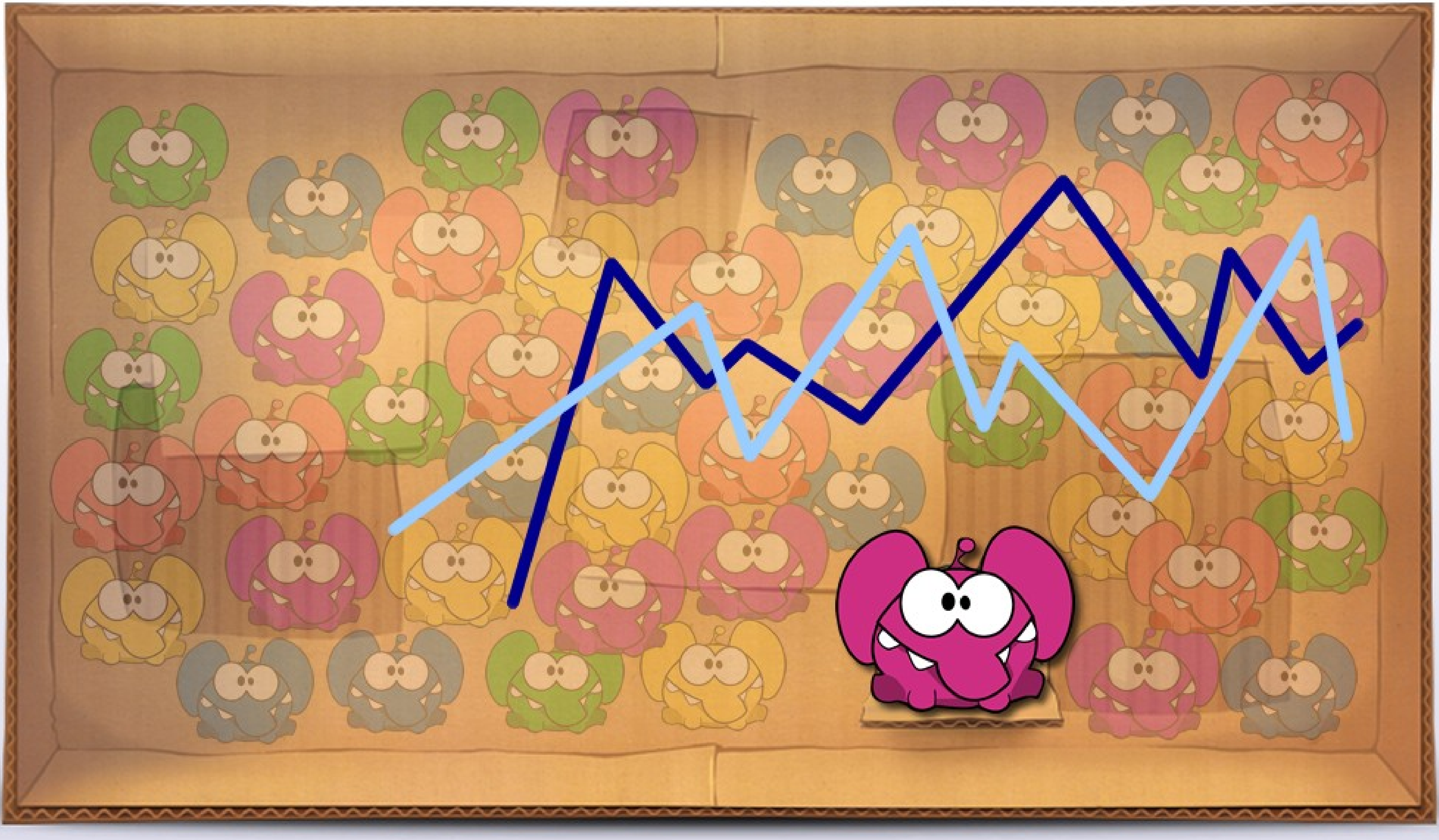
Erika Heidi

QUESTIONS

# THANKS!!!

erikaheidi.com/vagrant

joind.in/10712

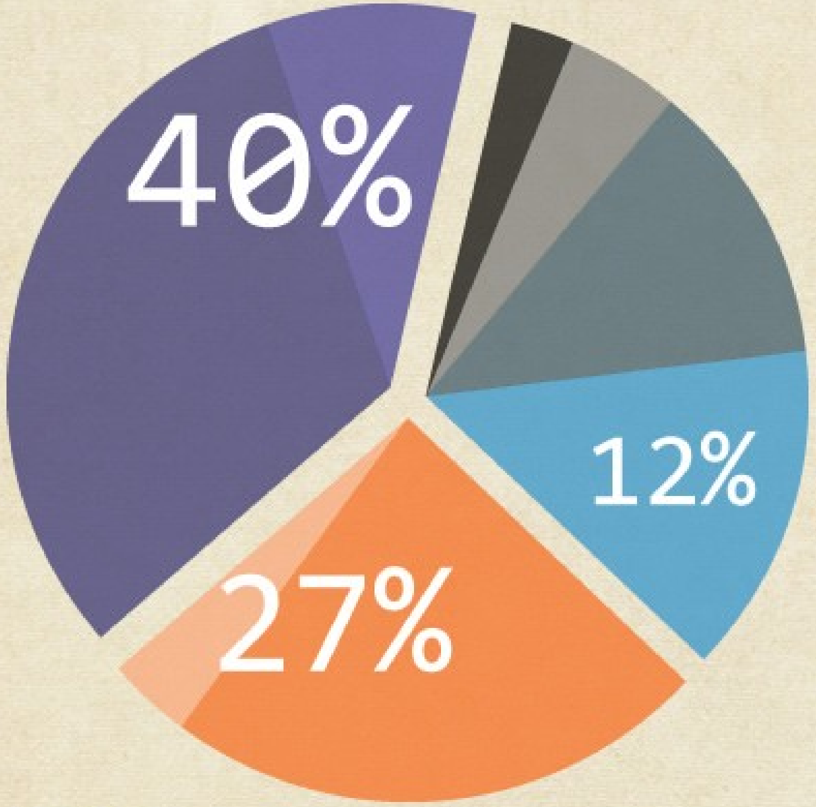


## Vagrant usage research

VAGRANT

USAGE RESEARCH
JANUARY 2014

# PROVISIONERS



| | | |
|---|---|---|
| 🟪 | puppet apply | 31% |
| 🟪 | puppet agent | 9% |
| 🟧 | chef solo | 23% |
| 🟧 | chef client | 4% |
| 🟦 | ansible | 12% |
| ⬛ | salt | 3% |
| ⬜ | shell | 6% |
| ⬛ | others | 12% |

PUPPET

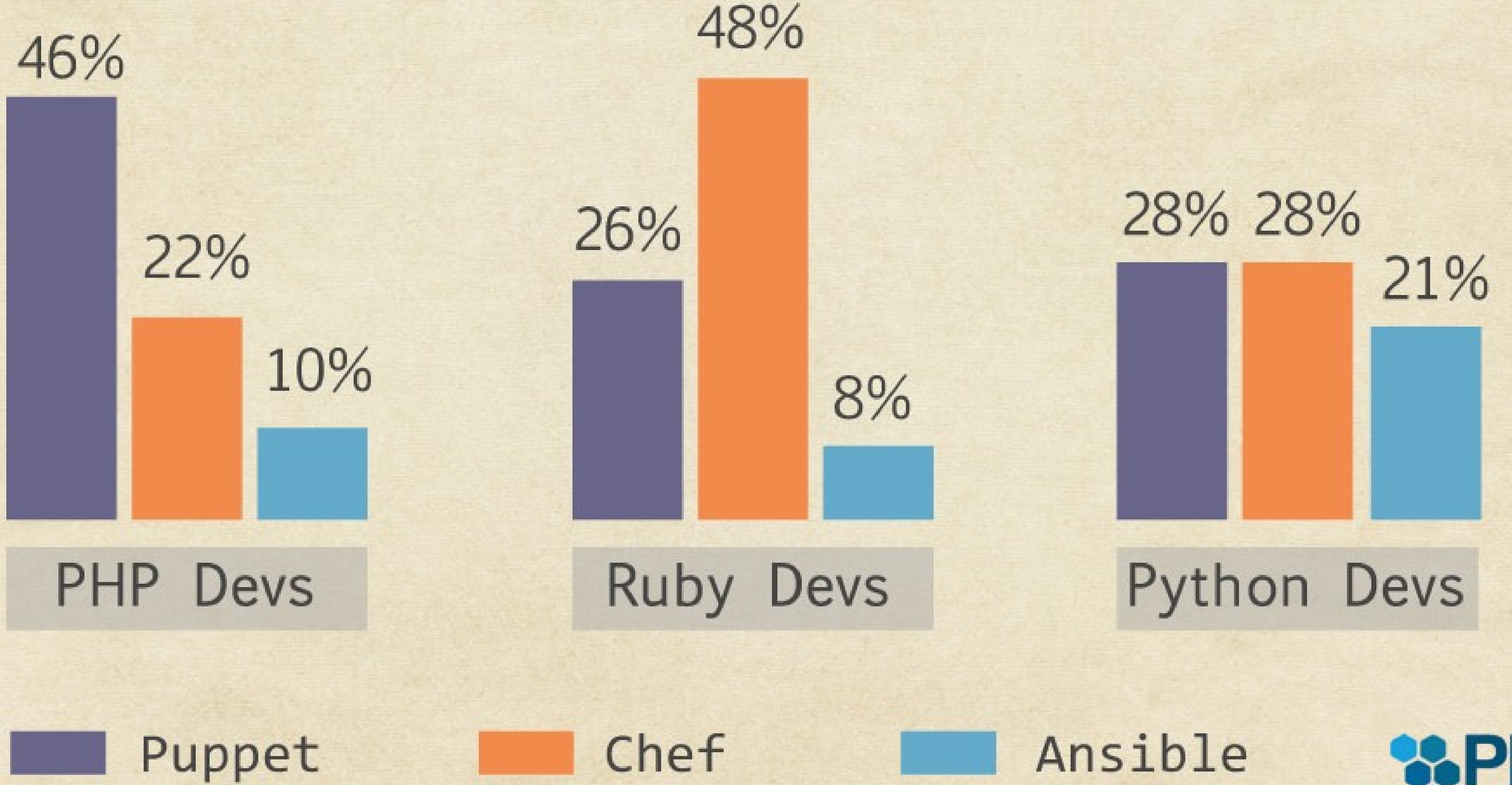
is the most popular
with 40%
(apply+agent)

PHP BENELUX 2014

USAGE RESEARCH
JANUARY 2014

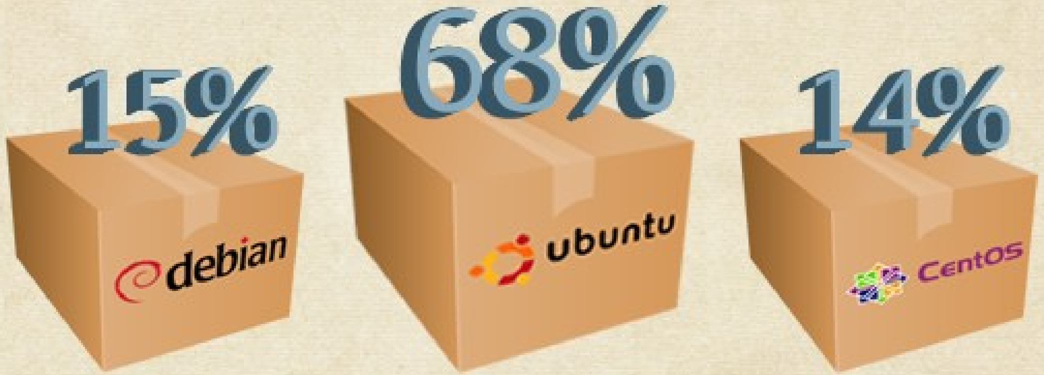
# PROVISIONERS

## ...except with Ruby developers.



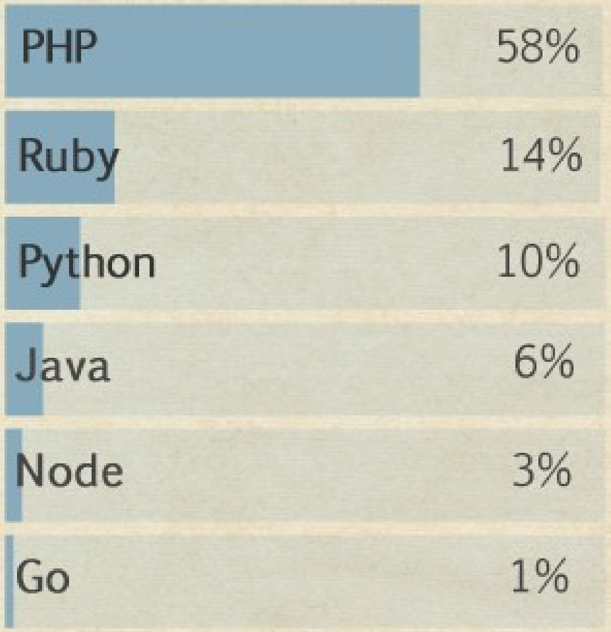
VAGRANT USAGE RESEARCH
JANUARY 2014

## BOXES



**58%** of boxes run apps written in

PHP



PHP BENELUX 2014

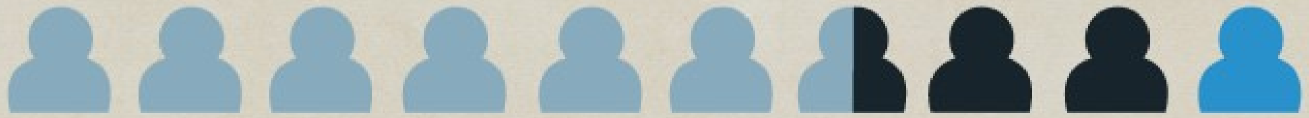
# USAGE RESEARCH

JANUARY 2014

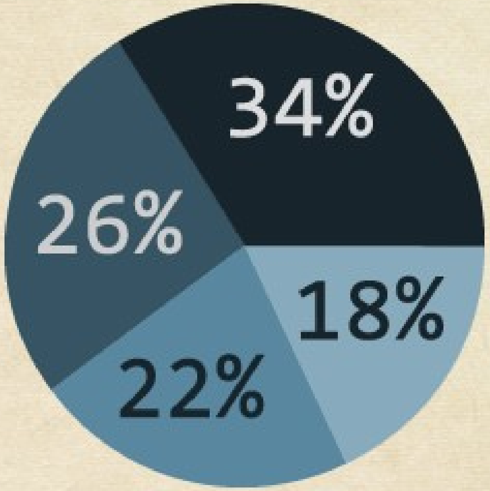
## USERS

VAGRANT

### 65% are OSX Users



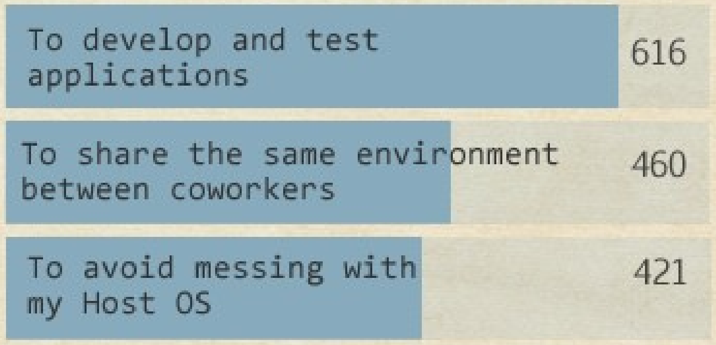
🟦 OSX Users ⬛ Linux Users 🟦 Windows Users



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?



PHP BENELUX 2014