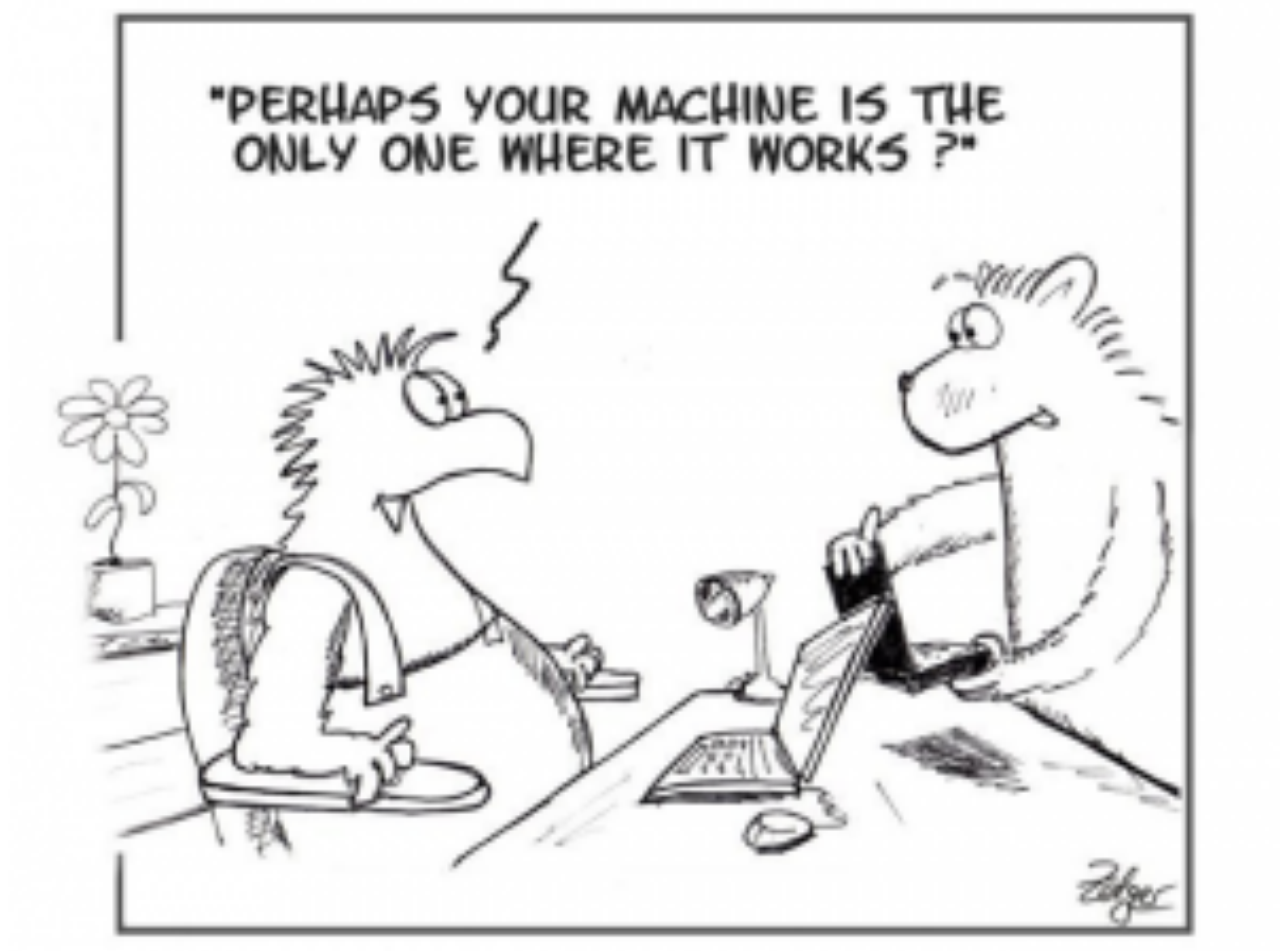
Tools: Vagrant

Designing and Maintaining Software (DAMS)

Louis Rose

Problem: “It works on my machine”

Bugs that appear in production and that can’t be reproduced by a developer on their machine are really hard to fix.

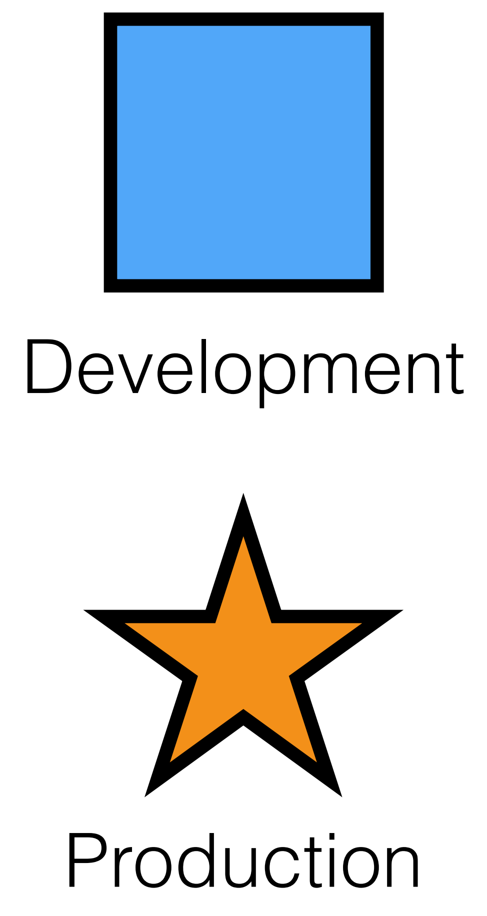


It works on my machine

Why does this happen?

Development and production environments are different:

- Different operating systems, compilers, etc?
- Different external services (fakes in development)?
- Different requirements:
  - Development needs debugging tools, fast tests, etc
  - Production needs better performance, security, etc



Solution: use virtual machines

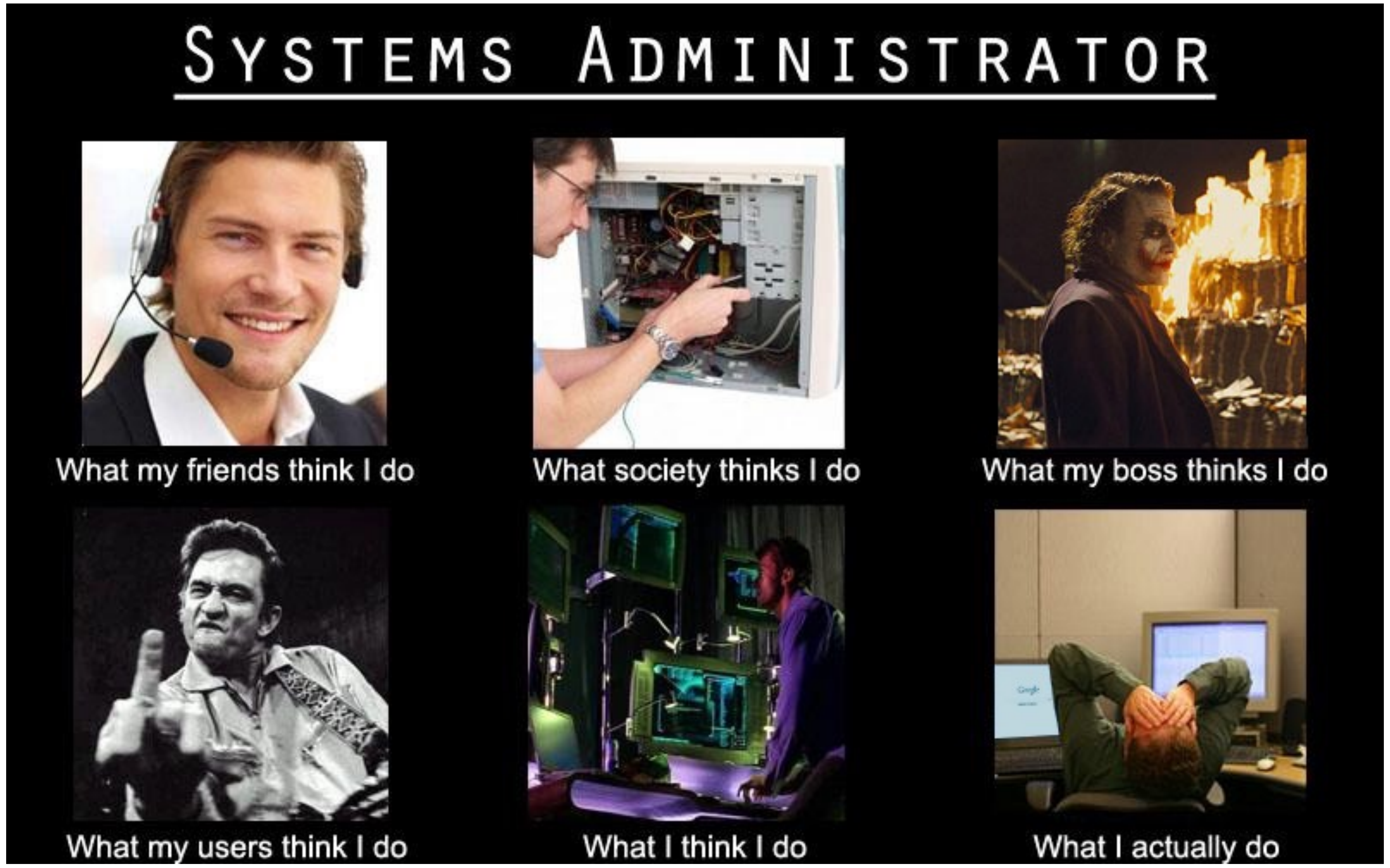
Developers can use a virtual machine that closely emulates the production environment.

- Same operating systems, compilers, etc.
- Same services
  (though some external services might need to be fakes)
- Different requirements:
  - Host machine has debugging tools
  - Guest (virtual) machine has production environment

Development
Production

Problem: VMs can be hard to setup

A production environment probably requires considerable sysadmin skills to setup correctly



Solution: Vagrant (or similar)

A tool for managing virtual machines

- Configuration is reproducible and sharable
- Base images are consistent
- Any additional setup can be automated
- Can run VMs locally or in the cloud (e.g., AWS)

Reproducible, Shareable Config

Every project can define one or more VMs in a Vagrantfile, which defines at least a base image and a provider.

```
Vagrant.configure("2") do |config|
  config.vm.box = "ubuntu/trusty64"

  config.vm.provider "virtualbox" do |vb|
    vb.memory = "1024"
  end
end
```

Reproducible, Shareable Config

Every project can define one or more VMs in a Vagrantfile,
and can also define networking and provisioning step.

```
Vagrant.configure("2") do |config|
  config.vm.box = "ubuntu/trusty64"

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

  config.vm.network "forwarded_port", guest: 80, host: 4567
  config.vm.provision "shell", path: "config/provision.sh"
end
```

Consistent base images

Vagrant provides a repository of open-source base images (“boxes”). Everyone gets the same base OS.

```
Vagrant.configure("2") do |config|
  config.vm.box = "ubuntu/trusty64"

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

  config.vm.network "forwarded_port", guest: 80, host: 4567
  config.vm.provision "shell", path: "config/provision.sh"
end
```

Automated provisioning

Vagrant can automatically run additional configuration steps such as shell scripts. This automates creating VMs.

```
Vagrant.configure("2") do |config|
  config.vm.box = "ubuntu/trusty64"

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

  config.vm.network "forwarded_port", guest: 80, host: 4567
  config.vm.provision "shell", path: "config/provision.sh"
end
```

Supports different providers

VMs can be local (via Virtualbox), or…

```
Vagrant.configure("2") do |config|
  config.vm.box = "ubuntu/trusty64"

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

  config.vm.network "forwarded_port", guest: 80, host: 4567
  config.vm.provision "shell", path: "config/provision.sh"
end
```

Supports different providers

VMs can be local (via Virtualbox), or in the cloud (e.g., via AWS).

```
Vagrant.configure("2") do |config|
  config.vm.box = "ubuntu/trusty64"

  config.vm.provider :aws do |aws, override|
    aws.access_key_id = "YOUR KEY"
    aws.secret_access_key = "YOUR SECRET KEY"
    aws.session_token = "SESSION TOKEN"
    aws.keypair_name = "KEYPAIR NAME"

    aws.ami = "ami-7747d01e"

    override.ssh.username = "ubuntu"
    override.ssh.private_key_path = "PATH TO PRIVATE KEY"
  end

  config.vm.network "forwarded_port", guest: 80, host: 4567
  config.vm.provision "shell", path: "config/provision.sh"
end
```

Summary

VMs can be used to increase the similarities between production and development environments, and reduce “works on my machine” problems.

Vagrant can be used to make it easier to create VMs, and to share the configuration of VMs between developers and sysadmins.