;login: logout

Setting Up a Vagrant Workflow

MATT SIMMONS

Matt Simmons is a 12+ year system administrator who works at the College of Computer and Information Science at Northeastern University in Boston. He blogs at http://www.standalone-sysadmin.com/ and can be reached via @standaloneSA on Twitter. standalone.sysadmin@gmail.com

Fire. The wheel. The Internet. Microwave cheese. These are things that we, as a species, have created and really matter to us in our daily lives, and that have appreciably made the world a better place. I want to cast my vote to add Vagrant [1] to that list.

Remember the bad old days, when you would write configuration management code, commit it to a repo, check it out in the testing environment, reboot a machine, and then a few minutes later figure out that you left out a semicolon, so you'd have to do it all over again? That whole workflow is so 2011.

Not that long ago, I was listening to some trainers talking about offering a Vagrant Box to people attending their classes at conferences. Being the naturally inquisitive sort of person that I am, I rudely interrupted their conversation to ask what they were talking about. I learned that Vagrant was apparently a "thing" that made "VMs" from "images".

Now, I'm more than passingly familiar with the whole "virtualization" deal, so I felt like I had a decent grasp of things from that description. I mean, I didn't think Vagrant was anything revolutionary, but I could kind of see where it was going. I thought Vagrant was something maybe like VMware's marketplace or maybe a nicer way for people to distribute their images or something. In a sense, I was kind of right, but in reality, I was way off. Vagrant is so much cooler than that.

As I found out later, Vagrant is an abstraction layer above virtualization solutions, typically things such as VirtualBox or VMware Fusion. These virtualization products, meant to be desktop solutions, have rather robust back-end capabilities and offer headless solutions that are of absolutely no use to you as a desktop, and they default to a console display if you use their interfaces, which is annoying if you want to use them as a server environment.

Vagrant is a way of automating and controlling the creation and destruction of those machines, but above and beyond that, Vagrant images (or boxes, in the lingo) have certain software installed, configured, and ready to be put to use by you for all of your nefarious (or not) purposes.

Here's how my current workflow looks. Suppose I've got an Ubuntu machine on which I want to play with Vagrant. I'll install it like this:

```
$ sudo install vagrant
```

By default, Vagrant doesn't come with any boxes to make new machines from, so lets add one:

```
$ vagrant box add precise32 http://files.vagrantup.com/precise32.box
```

References

[1] Vagrant: http://www.vagrantup.com/

[2] VagrantUp.com: http://www.vagrantup.com/

[3] Vagrant documentation: http://docs.vagrantup.com/v2/

Setting Up a Vagrant Workflow

This adds a box named precise32, and the source image…err, box…is downloaded from the given URL. We're really close now:

```
$ vagrant init precise32
```

This creates a configuration file that Vagrant will use to build the machine. The defaults will give us a nice clear template to work with. Now, we're ready:

```
$ vagrant up
```

Ta-Da! You now have a machine. You can connect to it like this:

```
$ vagrant ssh
```

It's up and running, with whatever image you wanted. Want to shut it down? Exit from the ssh session just like you normally would, then type:

```
$ vagrant destroy
```

Poof. Gone!

I realize that this is a simplistic example of what's possible, but look through the Vagrantfile and you'll see an entirely new world open before your eyes. You can create a Puppet or Chef configuration and have it run automatically on boot, or run an initialization shell script, or even create multiple VMs at once and build an entire infrastructure in miniature, then destroy it with less effort than it takes to kick down a sandcastle.

To check out Vagrant, I recommend working through the exercises at VagrantUp.com [2] and then read the documentation [3]. Vagrant has completely changed the way that I test my Puppet code, and once you grok it, I'm certain that it'll change yours, too. Feel free to write me at standalone.sysadmin@gmail.com and let me know what you think of it.