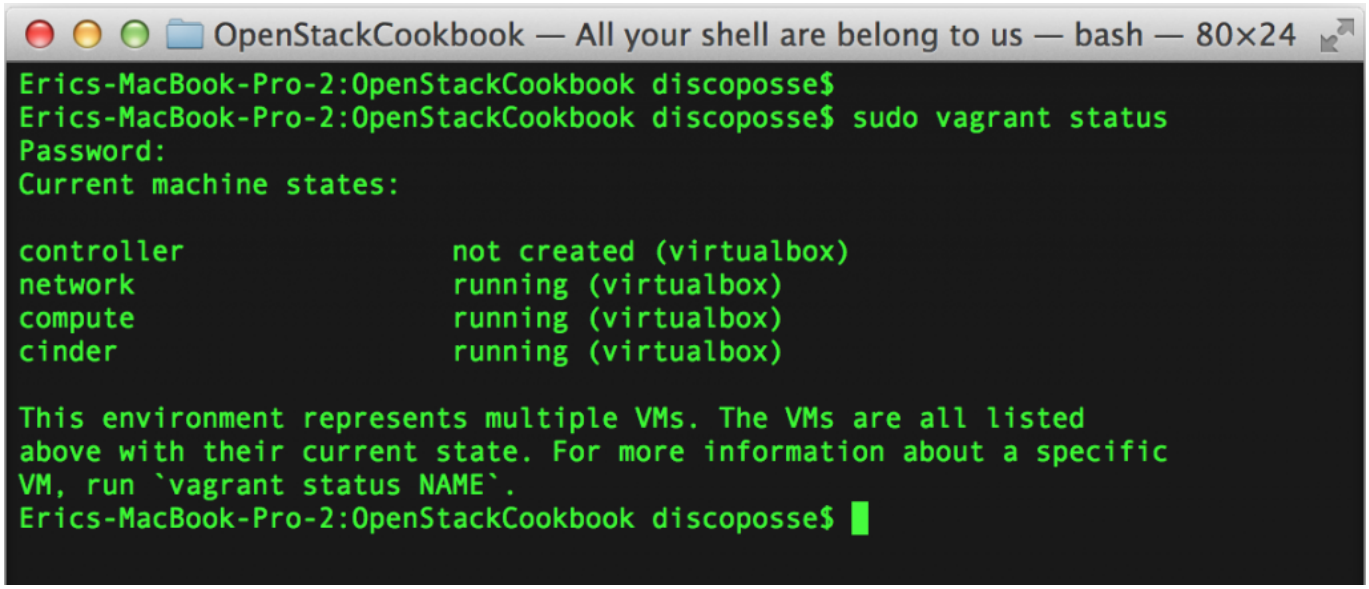
Suspend and Resume Vagrant Boxes in Bulk in OSX and Linux

As a regular user of Vagrant in conjunction with VirtualBox for building and deploying lab environments, I often have situations where I have multiple virtual instances running at a time. The challenge with that is that sometimes my multi-node labs are having to be suspended and resumed to save on resources. It's not difficult, but it does take multiple commands to complete the task.

There Has to be a Better Way!

Just like the infomercials always say: "There has to be a better way!", and in this case there absolutely is. Let's use an example where I have my OpenStack 4-node lab running in Virtual box. Because I'm security conscious, I also use sudo for all of my processes (and you should too), so let's just have a look at the process I go through to view, suspend and resume my 4-node lab with relative ease.

First, we want to list our vagrant boxes on the system: `sudo vagrant status`



We can see that we have a list of all of the nodes which shows them by name and with the status showing as **running**. In fact, I even have one machine that I haven't deployed that is shown as **not created**, which means that we will have even more confusion when trying to work with bulk commands potentially.

In order to work with our running machines only, we can simply pipe the output to a grep command to filter the results:

```
vagrant status | grep running
```

Now we have a fairly good bit of data to work with, but the problem is that our `vagrant suspend` command needs to use the instance name to control the VM. This means that we will need to pull one of the columns of data to pipe it to vagrant to issue the suspend command. So, let's do that!

Getting AWK-ward: Finding Text in the Stream

Because our last command produced a table of data which is able to be read and manipulated, we will use the `awk` command to print the first part of the data stream which is coming in.

```
sudo vagrant status | grep running | awk '{print $1}'
```

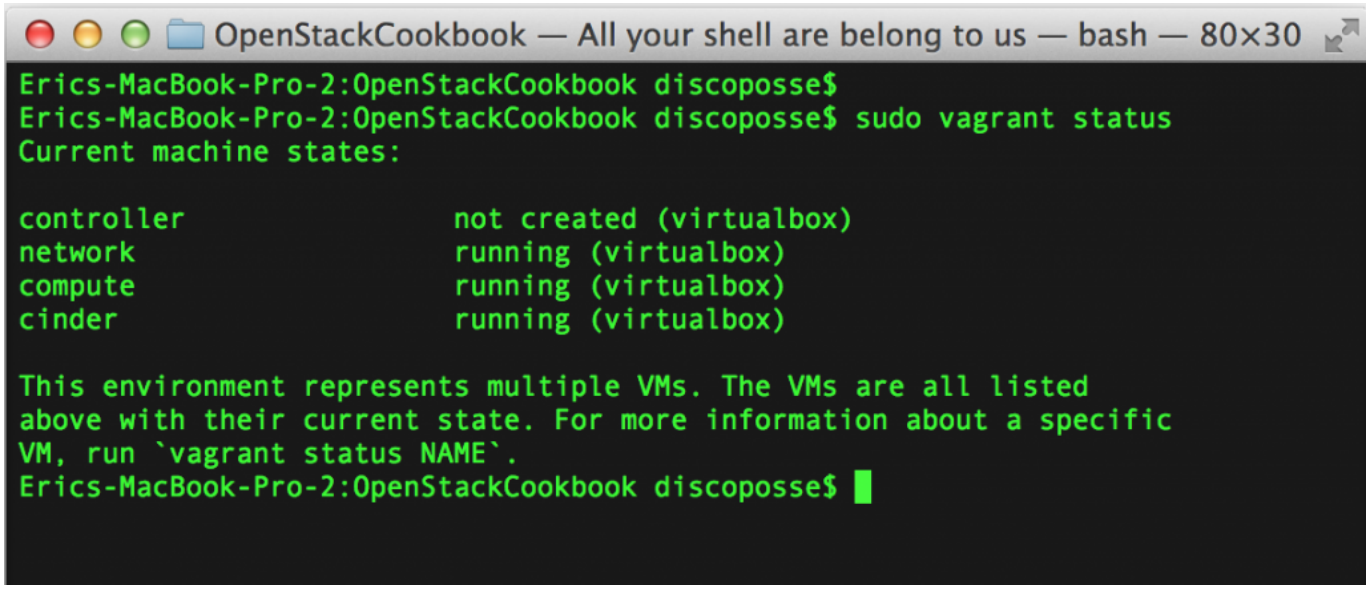
With this new set of data, we have exactly what we need to pass the output into a `vagrant suspend` command in order to suspend our virtual machines. In order to do this, we just add another pipe command into the one-liner:

```
sudo vagrant status | grep running | awk '{print $1}' | sudo vagrant suspend
```

Now to resume our machines, we simple revers the process. By checking the status and finding machines which are in the saved state, we can issue a resume command using this relatively simple one-liner:

```
sudo vagrant status | grep saved | awk '{print $1}' | sudo vagrant resume
```

Now you can run the vagrant status command and see that your virtual machines are up and running again:



This is a handy tip, especially when you're on battery and need to quiesce your lab machines to save processing power. Even when they seem idle, they are using additional resources. Beyond that, it's a great way to start working with the awk command and learn to start parsing data with it. As you will see with my OpenStack install scripts, awk is a powerful tool for dynamically parsing and piping data.

Hope you find this helpful!