Using Puppet + Vagrant to provision VMs

If you want to use Vagrant for testing your applications and have some desired initial state, using Puppet may be a choice to consider.

1. Prepare your Vagrant config

⚠ If you are not familiar with Vagrant yet, This link may be a good place to start.

Create a directory for your Vagrant config, and there you can create template `Vagrantfile` with:

```
vagrant init '<your_box_name>'
```

⚠ Choosing the right box to use is very important from the very beginning: you should make sure that pre-installed Puppet version will fit your needs

2. Enable puppet provisioning in the Vagrantfile

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

The location of the puppet manifest to be used for provisioning is `./manifests/default.pp`

There are some settings that may be used to change manifest dir, name, etc. Check Vagrant docs for the details.

Now you are done with minimal Vagrant configuration and can move to writing puppet manifest.

3. Write a manifest to be used for provisioning

By default the location of the manifest will be `./manifests/default.pp`

3.1. Choose puppet modules to use with your manifest.

⚠ The base of every good puppet manifest is so-called modules (puppet libraries) which provide a convenient way to describe system resources of specific types.

The set of the libraries used, obviously, depends on what kind of resources you want to manage. For example, if you want to setup rvm on the system, you should use some rvm-managing module, for example, `maestrodev/rvm`.

⚠ Be careful when choosing the modules: they may have restrictions on the puppet version installed on the guest OS.

The modules should be included in your config. The modules location is defined in config and, in our case, is `./modules/`.
There are two ways to install puppet modules:

1. Install them manually, but you would need to cope with dependencies as well;
2. Use `puppet module install` command which handles dependencies automatically. There are some parameters to use,

for example, `--target-dir`.

3.2 Write own manifest to describe the system.

For example, if you are using `maestrodev/rvm` module, one may use the following simple way to ensure some rubies are installed on the system:

```
include rvm

rvm::system_user {
  vagrant:
    create => false;
}

rvm_system_ruby {
  'ruby-1.9.3':
    ensure      => 'present',
    default_use => true,
#    build_opts  => ['--binary'];
}

rvm_gem {
  'bundler':
    name         => 'bundler',
    ruby_version => 'ruby-1.9.3',
    ensure       => latest,
    require      => Rvm_system_ruby['ruby-1.9.3'];
}
```


Using Puppet plugin for IntelliJ may help figuring out which resources does the module provide and which parameters do they have

4. Try your configuration

Use `vagrant up` to start your VM with provisioning.

Running puppet manifests may take quite a time so be patient.

See attachment for the example.

File	Modified
rvm_puppet.zip The sample configuration (Vagrant file, manifest, modules) for creating VM with rvm.	Apr 13, 2015 by Valentin Fondaratov