Languages and Algorithms for Distributed Applications (LSINF2345) - Lab Session 1

# 1. Background

In this lab session, you will use some well-known tools for automatically creating and provisioning development environments, namely Vagrant and Puppet.

Vagrant is used to create and manage Virtual Machines (VMs). Once we have a VM running, we will use another tool – called Puppet – to automatically deploy software and configurations on the VM.

Vagrant and Puppet both use configuration scripts to know how to provision the VM. We’ll supply you with pre-defined scripts to generate a VM that will serve as an initial baseline for your riak cluster.

# 2. Provisioning the VM with Vagrant and Puppet

**Note**: These instructions assume you are working on a machine of your own, where you have permission to install new software. If you can only access the machine in the computer rooms, we can supply you with a pre-made VM to be run there.

1. Install Vagrant from <https://www.vagrantup.com/downloads>
2. Install Puppet from <https://docs.puppetlabs.com/guides/install_puppet/pre_install.html#next-install-puppet>  
   You don’t need Facter or Hiera.
3. Install VirtualBox from <https://www.virtualbox.org/wiki/Downloads>
4. Clone the course repository by running  
   git clone <https://github.com/wreda/lsinf2345-project>  
   If you don’t have git, you can install it from <http://git-scm.com/downloads>.

The vm directory of the repository contains these files:

* vagrantfile: The configuration used by Vagrant to create the VM: memory, network interfaces, provisioning tools – such as puppet – to use, …
* manifest/base.pp: The script describing the configuration and software to be installed on the VM by Puppet.

1. Run vagrant up in the vm directory. This will create the VM, launch it, then run Puppet to configure it and install all required software. Do not attempt to use the VM until the Vagrant yields the shell to you on the host OS.
2. Log into the VM. The default login and password are both “vagrant” (beware the keyboard layout of the login screen).

Note that the vm serves as a shared directory between the VM and your machine. On the VM, this directory is mounted under /vagrant.

# Appendix A: Additional Notes on Vagrant and Puppet

You can use the following Vagrant commands to clean up your environment:

* vagrant suspend: Will save the current machine’s state and stop it. Requires more disk space but allows you to resume work faster
* vagrant halt: Equivalent to a graceful shutdown. Takes more time to start from a cold boot, and the VM still consumes space (albeit lower than the previous command)
* vagrant destroy: Will delete the guest machine from your system but will keep your box. VM itself will consume no diskspace at the cost of having to restart the provisioning process if you want to rebuild your VM.

You can try to modify the puppet script, if you want to automate other configuration changes to your VM. Remember however that the original script is what we use as our baseline.

Take a look at the puppet language tutorial here:<https://docs.puppetlabs.com/puppet/latest/reference/lang_summary.html>

A few other notes for the pedantic readers:

* In order to apply any puppet script changes to your VM you can use the command: vagrant reload --provision
* If you plan to apply Puppet configuration changes to your VM, take a snapshot of the VM via Virtualbox or through a vagrant plugin: <https://github.com/dergachev/vagrant-vbox-snapshot>