Chapter 4

Lab 4.2

Objective: Deploying a Java application with Puppet infrastructure automation

Introduction to Puppet

Puppet, like Chef, is fundamentally a client-server based configuration management system. Like the previous Chef example, this exercise will focus on using Puppet in a single-system mode to deploy one server.

This lab will demonstrate how to write a single Puppet manifest which uses Ubuntu's package management to deploy the Java application. This is a different approach from the Chef example. It is far simpler, but with the disadvantage of not easily being able to deploy specific versions of Java or Tomcat.

Using Puppet to Deploy Your Application

Step 1. Create a new Ubuntu virtual machine, and, as a root, install Puppet's client tools:

1. Use Ubuntu's package manager to install Puppet:

```
$ sudo apt-get install puppet
```

Step 2. Initialize the Puppet local directory as a Git repository for easy delivery:

1. Change the directory to the local Puppet directory:

```
$ cd /etc/puppet
```

2. Use a text editor to ensure that `/etc/puppet/puppet.conf` has the following contents:

   ***Note**: The `[master]` section is not necessary because we are running in standalone mode.*

```
[main]
logdir=/var/log/puppet
vardir=/var/lib/puppet
ssldir=/var/lib/puppet/ssl
rundir=/var/run/puppet
factpath=$confdir/facter
```



```
prerun_command=/etc/puppet/etckeeper-commit-pre
postrun_command=/etc/puppet/etckeeper-commit-post
```

3. Initialize this directory as a Git repository:

```
$ git init
```

4. Track all of the Puppet files in Git:

```
$ git add
```

5. Commit these files to your new repository:

```
$ git commit -m "initial commit"
```

Step 3. Create a module for your `petclinic` installation:

***Note**: You may need `sudo` privileges to create and/or edit some of the following files depending on your directory permissions*.

1. Navigate to the Puppet **`modules`** directory:

```
$ cd/etc/puppet/modules
```

2. Create directories for the application files, as well as the Puppet manifest:

```
$ mkdir -p petclinic/manifests petclinic/files
```

3. Navigate to the **`manifests`** directory:

```
$ cd petclinic/manifests
```

Step 4. Create an `init.pp` file in the `petclinic/manifests` directory and add the following contents:

```
  java

$ class petclinic {

$ exec{'apt-update':
  command => '/usr/bin/apt-get update'
  }

$ packages = ['openjdk-7-jre', 'tomcat7']

$ package { $packages:
  require => Exec['apt-update'],
```



```
  ensure => installed,
  }

$ file {'/usr/share/tomcat7':
  owner => 'tomcat7',
  group => 'tomcat7',
  ensure => directory,
  require => Package['tomcat7'],
  }

$ file {'/var/lib/tomcat7/webapps/petclinic.war':
  owner => 'tomcat7',
  group => 'tomcat7',
  mode => '0644',
  ensure => present,
  require => Exec['download-petclinic-war']
  }

$ exec { 'download-petclinic-war':
  command => '/usr/bin/curl -o petclinic.warhttp://your.artifactory.ip:80
  81/artifactory/path/to/petclinic/petclinic-1.0.0-SNAPSHOT.war',
  creates => '/var/lib/tomcat7/webapps/petclinic.war',
  cwd => '/var/lib/tomcat7/webapps',
  require => Package['tomcat7']
  }

$ service { tomcat7:
  ensure => running,
  enable => true
  }

 $ exec{'add-tomcat-java-home':
   cwd => ['/etc'],
   path => ['/etc','/usr/bin'],
   command  =>  '/bin/echo  "JAVA_HOME=/usr/lib/jvm/java-7-openjdk-amd64/"  >>
   /etc/default/tomcat7',
   notify => Service['tomcat7']
   }

   }
```

Step 5. Create `/etc/puppet/manifests/site.pp` with the following contents:

```
java
node default {
```



```
include petclinic
}
```

The system can be tested by running the following command:

```
$ puppet apply /etc/puppet/manifests/site.pp
```

Step 6. After successfully deploying your application with Puppet, finalize the Git repository:

1. Change to the repository's root:

   ```
   $ cd /etc/puppet
   ```

2. Add the new files to SCM:

   ```
   $ git add
   ```

3. Commit changes:

   ```
   $ git commit -m "Added petclinic module and site.pp"
   ```

This repository can be added to Github or any other Git repository management system. It is possible to now create a simple script to deploy the PetClinic application to a new node. An example is provided here:

```
#!/bin/bash

$ apt-get update
$ apt-get install -y update
$ cd /etc
$ rm -rf puppet
$ git clone https://github.com/your/puppetrepo
$ mv puppetrepo puppet
$ puppet apply /etc/puppet/manifests/site.pp
```

Note: You will need to provide authentication if this is a private repository. You can create a Puppet module to run `puppet apply` in the local `crontab` to keep things up-to-date as well.

