An Introduction to Puppet Enterprise

Exercise & Lab Guide

Puppet Education
www.puppetlabs.com/education

Lab 3.1: Pre-installation

Objective:

Assign a hostname to your agent and make that name persist across reboots

Steps:

Edit your system host file

1. Log in to your virtual machine with the username "*root*" and password "*puppet*"
2. Note the system's IP address displayed; if you missed it type `ifconfig eth0`
   - Your agent's IP _____
3. Ask your instructor for the classroom master's IP address
   - The master's IP _____
4. Ensure your VM's clock is in sync with the Puppet Master
   - `ntpdate us.pool.ntp.org`
5. Choose a username for class consisting of alphanumeric characters only. It will be used throughout the course and referred to as *yourname*.
   - Your username _____
6. Edit the system host file and add entries for `master`, `puppet`, and *yourname*
   - `vim /etc/hosts`
   - append "*the-masters-ip* master.puppetlabs.vm master puppet"
   - append "*your-agents-ip* **yourname**.puppetlabs.vm *yourname*"

Example File:

```
# Do not remove the following line, or various programs
# that require network functionality will fail.
127.0.0.1   localhost.localdomain   localhost
::1 localhost6.localdomain6 localhost6
192.168.X.X master.puppetlabs.vm master puppet
192.168.Y.Y yourname.puppetlabs.vm yourname
```

continued...

Configure your system hostname

1. Set your hostname for the current session and to be persistent across reboots
   - `hostname yourname.puppetlabs.vm`
   - `exec bash`
   - `vim /etc/sysconfig/network`

Example File:

```
NETWORKING=yes
NETWORKING_IPV6=yes
HOSTNAME=yourname.puppetlabs.vm
```

Validate your work.

1. Use the validation script we have provided to check your work on each step above.
   - `verify_fundamentals_settings.sh` **yourname**

Expected Results:

```
[root@yourname ~]# verify_fundamentals_settings.sh yourname
Checking hostname                                          [  OK  ]
Checking hostname validity                                 [  OK  ]
Checking that the hostname will be set on boot             [  OK  ]
Checking master name resolution                            [  OK  ]
Checking local hostname resolution                         [  OK  ]
```

Please note that later exercises will use `training` as the student environment name, hostname, etc.

Lab 3.2: Installation

Objective:

Install Puppet Enterprise on your virtual machine.

Steps:

Install Puppet Enterprise on the virtual machine by running the installer located in the `puppet-enterprise` directory.

Create an answer file for your Puppet Enterprise installation

1. Create an answer file for your installation
   - `cd ~/puppet-enterprise/`
   - `./puppet-enterprise-installer -s answers.txt`
     - Install both the `agent` and the `master` role.
     - Press `[enter]` to accept most default answers.
     - When asked for the Admin email address for accessing the console interface, use **yourname**@puppetlabs.vm`.
     - When asked for the Password for user **yourname**@puppetlabs.vm, use `puppetlabs`
     - When asked for the SMTP server to email account information to users, use **yourname**.puppetlabs.vm
     - For everything else, press `[enter]` to accept default answers.

2. Review the answer file that was just created and ensure there are no mistakes
   - `cat answers.txt`

Example File:

```
q_install=y
q_puppet_cloud_install=n
q_puppet_enterpriseconsole_auth_database_name=console_auth
q_puppet_enterpriseconsole_auth_database_password=V2MKgtv7BwT2LPIqICTQ
q_puppet_enterpriseconsole_auth_database_user=console_auth
q_puppet_enterpriseconsole_auth_password=puppetlabs
q_puppet_enterpriseconsole_auth_user_email=yourname@puppetlabs.vm
q_puppet_enterpriseconsole_database_install=y
```



```
q_puppet_enterpriseconsole_database_name=console
q_puppet_enterpriseconsole_database_password=iHEYmzukC6pNGoiH1DV9
q_puppet_enterpriseconsole_database_remote=n
q_puppet_enterpriseconsole_database_root_password=WbT6LmyKkYLXkugZ26e6
q_puppet_enterpriseconsole_database_user=console
q_puppet_enterpriseconsole_httpd_port=443
q_puppet_enterpriseconsole_install=y
q_puppet_enterpriseconsole_inventory_hostname=yourname.puppetlabs.vm
q_puppet_enterpriseconsole_inventory_port=8140
q_puppet_enterpriseconsole_master_hostname=yourname.puppetlabs.vm
q_puppet_enterpriseconsole_smtp_host=yourname.puppetlabs.vm
q_puppet_enterpriseconsole_smtp_password=
q_puppet_enterpriseconsole_smtp_port=25
q_puppet_enterpriseconsole_smtp_use_tls=n
q_puppet_enterpriseconsole_smtp_user_auth=n
q_puppet_enterpriseconsole_smtp_username=
q_puppet_symlinks_install=y
q_puppetagent_certname=yourname.puppetlabs.vm
q_puppetagent_install=y
q_puppetagent_server=yourname.puppetlabs.vm
q_puppetca_install=y
q_puppetmaster_certname=yourname.puppetlabs.vm
q_puppetmaster_dnsaltnames=puppet,puppet.puppetlabs.vm,yourname,yourname.puppetlabs.v
q_puppetmaster_enterpriseconsole_hostname=localhost
q_puppetmaster_enterpriseconsole_port=443
q_puppetmaster_install=y
q_vendor_packages_install=y
q_verify_packages=y
```

continued...

Install Puppet Enterprise using the answer file you created

1. Perform your installation using the answer file you created
   - `cd ~/puppet-enterprise/`
   - `./puppet-enterprise-installer -a answers.txt`

Explore your new Puppet Enterprise installation

1. Run: `puppet -V`
   - *Please note the option above is a capital V*
   - should display the `puppet` agent's version
2. Run: `puppet agent --configprint confdir`
   - should display the `puppet` agent's configuration directory
     - `/etc/puppetlabs/puppet`
3. Run: `puppet agent --configprint certname`
   - should display the `puppet` agent's certificate name:
     - **yourname**`.puppetlabs.vm`
4. Run: `puppet agent --configprint server`
   - should display the `puppet` agent's master:
     - **yourname**`.puppetlabs.vm`

Exercise 3.3: Facter

Objective:

Become familiar with the use of facter

Steps:

1. Execute `facter ipaddress`
   - What is returned?
2. Execute `facter operatingsystem`
   - What is returned?
3. Execute `facter`
   - What is returned?

Discussion Questions

- What sort of information is returned by facts?
- How might you use Facter outside of Puppet?
- Why do you think Facter exposes facts such as `ipaddress_eth0` instead of a single array of values?

Exercise 3.4: Puppet Resource

Objective:

Use `puppet resource` to inspect user accounts

Steps:

Create a user account on your system

1. Create your user account manually with `useradd`
   - `useradd -s /bin/bash -b /home` **yourname**
2. Using puppet resource, inspect your user account on the system
   - `puppet resource user` **yourname**
   - Note the "`password =>`" value

Add a password to your account

1. Set your new user account password to "*puppetlabs*"
   - `passwd` **yourname**
2. Using puppet resource, inspect your user account on the system
   - `puppet resource user` **yourname**
   - Note that the "`password =>`" value has changed

Discussion Questions

- How might you inspect other resource types, such as groups or files?
- Why does `puppet resource` return so much information?
- Do you think that `puppet resource` could list iptables rules if a resource type existed for them? Why or why not?

Lab 5.1: Build Your First Module

Objective:

Construct and test a Puppet Module to manage user account `introduction`.

Steps:

1. Ensure your current working directory is the *modulepath*
   - `cd /etc/puppetlabs/puppet/modules`
2. Create your manifests directory
   - `mkdir -p users/manifests`
3. Edit your manifest
   - `vim users/manifests/init.pp`
4. Use `puppet parser validate` to validate the syntax of your manifest
   - `puppet parser validate users/manifests/init.pp`

Expected result

```
[root@training modules]# puppet parser validate users/manifests/init.pp
[root@training modules]#
```

There will be no output if the syntax of your file is correct.

Discussion Questions

- Will `puppet parser validate` tell you if you use the wrong attribute name?
- Will `puppet parser validate` tell you if you misspell the resource type?
- Why or why not?

Lab 5.2: Use Your Module

Objective:

Enforce your `users` class on your local agent.

Steps:

1. Ensure your current working directory is the *modulepath*
   - `cd /root/puppetcode/modules`
2. Create your tests directory
   - `mkdir users/tests`
3. Create your smoke test
   - `vim users/tests/init.pp`
4. Validate your syntax and simulate your smoke test
   - `puppet parser validate users/tests/init.pp`
   - `puppet apply --noop users/tests/init.pp`
5. Enforce your class on the local system
   - `puppet apply users/tests/init.pp`

Expected results

```
[root@training modules]# puppet apply --noop users/tests/init.pp
notice: /Stage[main]/Users/User[fundamentals]/ensure: current_value absent, should
notice: Class[Users]: Would have triggered 'refresh' from 1 events
notice: Stage[main]: Would have triggered 'refresh' from 1 events
notice: Finished catalog run in 0.28 seconds
[root@training modules]# puppet apply users/tests/init.pp
notice: /Stage[main]/Users/User[fundamentals]/ensure: created
notice: Finished catalog run in 0.37 seconds
```

Discussion Questions

- What role should your smoke tests play in your complete testing strategy?
- What limitations do you think that `--noop` mode might have?
- Can you identify a scenario in which the `--noop` run could succeed but the real run would fail?

Lab 5.3: Expand Your Module

Objective:

Extend your module to manage multiple resource types

Steps:

1. Ensure your current working directory is the *modulepath*
   - `cd /root/puppetcode/modules`
2. Edit your manifest
   - `vim users/manifests/init.pp`
3. Modify your `users` class to provide the following additional attributes
   - Add a group resource to manage the "*staff*" group
   - Add the "*staff*" group to `fundamentals`
   - Add the `/bin/bash` shell to `fundamentals`
4. Validate the syntax of your class and enforce it locally
   - `puppet parser validate users/manifests/init.pp`
   - `puppet apply users/tests/init.pp`

Expected results

```
[root@training modules]# puppet apply users/tests/init.pp
notice: /Stage[main]/Users/Group[staff]/ensure: created
notice: /Stage[main]/Users/User[fundamentals]/shell: shell changed '/bin/zsh' to '
notice: Finished catalog run in 0.28 seconds
```

Discussion Questions

- Why might you not see the shell change during this lab?
- What would happen if you used `chsh` to modify the user's shell and ran again?
- Can you set the shell to any arbitrary value? Why or why not?

Lab 5.1: Proposed Solution

Build Your First Module

Your module structure should resemble:

```
[root@training modules]# tree users/
users/
└── manifests
    └── init.pp
```

Example file: `users/manifests/init.p`

```
class users {
  user { 'introduction':
    ensure => present,
  }
}
```


Lab 5.2: Proposed solution

Use Your Module

Your module structure should resemble

```
[root@training modules]# tree users/
users/
├── manifests
│   └── init.pp
└── tests
    └── init.pp
```

Example file: `users/tests/init.p`

```
include users
```


Lab 5.3: Proposed Solution

Extend your module to manage multiple resource types

Your module structure should resemble

```
[root@training modules]# tree users/
users/
├── manifests
│   └── init.pp
└── tests
    └── init.pp
```

Example file: `users/manifests/init.p`

```
class users {
  user { 'fundamentals':
    ensure => present,
    gid    => 'staff',
    shell  => '/bin/bash',
  }

  group { 'staff':
    ensure  => present,
  }
}
```

