Lab 4.2: Creating a Puppet Module
IN719 Systems Administration

Introduction

A Puppet *module* is a collection of related resources that can be applied to a node as a single unit. For example, you may want to manage a service, together with its configuration files. In this lab we will create a simple module to manage `sudo`

Creating the module

Modules can be placed in few standard locations, typically `/var/lib/puppet/modules`, `/etc/puppet/modules`, and `/opt/modules`. Create your sudo module in `/etc/puppet/modules`. To begin, create the following directories, all under `/etc/puppet/modules`

```
sudo
sudo/files
sudo/templates
sudo/manifests
```

Also create the file `sudo/manifests/init.pp`. Every module must have this basic structure.

Next, edit the init.pp file so that it contains the following:

```
class sudo {
  package { 'sudo':
      ensure => present,
  }

  file { '/etc/sudoers':
      owner => "root",
      group => "root",
      mode => 0440,
      source => "puppet:///modules/sudo/etc/sudoers",
      require => Package['sudo'],
  }
}
```

We defined a class, “sudo” with two resources in it. The first makes sure that the *sudo* package is installed. The second directs Puppet to copy the configuration file we will place in the module. It also sets various properties on the copied file. The source directive says to take the file from `etc/sudoers`, which will be found under the `files` subdirectory of the module. The last directive, “require” directs Puppet to ensure that the sudo package is installed before copying the configuration file.

In the future we can make changes to the configuration file in the module and that updated configuration will be distributed to any agents that need it without requiring us to update the manifest or take any other action.

We still need to place a copy of a sudoers file in the module. Take a good copy of your sudoers file and copy it (to the file) `/etc/puppet/modules/sudo/files/etc/sudoers`. Ensure that is readable by the system user used by the puppetmaster service.

Note: The physical location (`/etc/puppet/modules/sudo/files/etc/sudoers`) and the instruction (`source => "puppet:///modules/sudo/etc/sudoers"`) as part of the module specification are slightly different. The module file assumes that all files are in the subfolder “files” and thus does not include it in the instruction. This is important to bear in mind whenever you write modules.

Applying the module to a node

Disclaimer: It is best to log on in a second console instance and elevate to root privileges, before you apply that module as described below - in case something goes wrong. This way you can still fix the sudoers file manually in case things don’t work out as planned.

Once the module is ready we can apply it to a node. To do this you only need to add one line to the `nodes.pp` file. In the node resource you created for the DB server in that last lab, add the line

```
include sudo
```

You can watch the agent apply the configuration by starting it manually:

```
sudo puppet agent --server=mgmt.op-bit.nz --no-daemonize --verbose --onetime
```

Remaining tasks

Using Puppet to configure one server is a bit of a drama. Get puppet agents running on your remaining Linux hosts, create node resources for them, and start creating modules for the resources that they have in common. Don't forget that the management server should also be an agent itself.

You want your Puppet agents to start on boot and run as daemons. Right now they are not configured to do that, but if you try to start the Puppet agent daemon, it basically tells you what you need to do. Get your Puppet agents running as daemons. Ensure you keep an eye on the log file to see track their activity (the log file directory is configured in the main configuration file).