**configure puppet server and puppent agent on ubuntu 18.04**

**Introduction**

* **Puppet is a configuration management tool that helps system administrators automate the provisioning, configuration and management of a server infrastructure. Planning ahead and using config management tools like Puppet can cut down on time spent repeating basic tasks and help ensure that configurations are consistent and accurate across your infrastructure.**
* **When you think of a configuration management tool, the one pop up in your mind is Puppet. Puppet is an open source configuration management tool, helps you to deploy and manage the configurations of hundreds of client systems from the central location.**
* **Puppet makes the system admin’s life easier by cutting down on time spending on repetitive task and allows them to work on other productive works, also ensures that all the configuration are consistent across the infrastructure.**
* **Puppet is available for Linux, Mac, BSD, Solaris, and Windows-based computer Systems, released under Apache License, written in “Ruby” language.**

**Environment**

**Here, we will configure a puppet in master/agent architecture and will use three systems, as mentioned below**.

**Puppet Master**

Host Name: server.mit.com

IP Address: 192.168.72.91

**Puppet clients**

Host Name: client.mit.com

IP Address: 192.168.72.92

Host Name: agent.mit.com (centos\_7)

IP Address: 192.168.1.72.94

**Puppet agent uses the hostname to communicate with the Puppet Server. So, make sure the agent machine can resolve the hostname of the Puppet Server.**

**On each machine, edit the /etc/hosts file. At the end of the file, specify the Puppet master server**

vi /etc/hosts

server.mit.com 192.168.72.91 server

client.mit.com 192.168.72.92 client

agent.mit.com 192.168.72.94 agent

**Puppet server is not available in Ubuntu 16.04 default repository. So you will need to add Puppet Lab repository on both Master and Agent machine.**

**On each machine, run the following command to download and install Puppet repository:**

**### Ubuntu 18.04 ###**

wget https://apt.puppetlabs.com/puppet6-release-bionic.deb

sudo dpkg -i puppet6-release-bionic.deb

sudo apt update

**### centos 7 ###**

rpm -Uvh https://yum.puppet.com/puppet6-release-el-7.noarch.rpm

yum update

**Configure Puppet Server**

**Puppet Server is the server software that runs on the puppet master machine. Install the Puppet server using below command.**

apt install -y puppetserver

**By default, Puppet Server is configured to use 2GB of RAM. However, if we want to experiment with Puppet Server on a VM, we can safely allocate as little as 512MB of memory. To change the Puppet Server memory allocation, we can edit the init config file, /etc/default/puppetserver:**

vi /etc/default/puppetserver

**(Change the value shown in red)**

**From:**

JAVA\_ARGS="-Xms2g -Xmx2g -Djruby.logger.class=com.puppetlabs.jruby\_utils.jruby.Slf4jLogger"

**To:**

JAVA\_ARGS="-Xms512m -Xmx512m -Djruby.logger.class=com.puppetlabs.jruby\_utils.jruby.Slf4jLogger"

**Simple Configurations**

**Puppet Server does not require any configuration, and you can simply start the puppetserver service. It will use the default settings.**

For ex: dns\_alt\_names server.mit.com

vi /etc/puppetlabs/puppet/puppet.conf

[master]

dns\_alt\_names = server.mit.com,server

[main]

certname = server.mit.com

server = server.mit.com

**Generate a root and intermediate signing CA for Puppet Server.**

**run below command**

/opt/puppetlabs/bin/puppetserver ca setup

*Output:*

*Generation succeeded. Find your files in /etc/puppetlabs/puppet/ssl/ca*

**Start and enable the Puppet Server.**

systemctl start puppetserver

systemctl enable puppetserver

**Install Puppet Agent**

**Now, your Puppet server is up and running. It’s time to install Puppet agent on Agent machine.**

**Before installing Puppet agent, make sure you have installed Puppet Lab repository on Agent machine. Next, install Puppet agent by just running the following command:**

apt-get install puppet-agent -y

yum install -y puppet-agent (centos\_7)

(note this procedure will same as it is on centos also)

**Once Puppet agent is installed, you will need to edit the puppet configuration file and set puppet master information.**

**You can do this with the following command:**

vi /etc/puppetlabs/puppet/puppet.conf

Add the following lines:

[main]

certname = client.mit.com

server = server.mit.com

**Save and close the file, then start Puppet agent service and enable it**

systemctl start puppet

systemctl enable puppet

**Sign the Puppet Agent Certificate on Puppet Server**

**When the Puppet runs Agent machine first time, it sends a certificate signing request to the Puppet server. In Client-Server architecture, Puppet master server must approve a certificate request for each Agent machine to control the Agent machine.**

**On Puppet server,**

**Log into the puppet master server and run below command, it will show all unsigned certificate requests**

sudo /opt/puppetlabs/bin/puppetserver ca list

*output:*

*client.mit.com (SHA256) 6F:E4:F2:FE:61:D7:91:0A:CB:0B:AB:33:21:0E:92:DC:21*

*agent.mit.com (SHA256) CA:B5:67:1B:CD:8B:72:16:45:C6:DD:7D:BE:1D:D7:CC:5C*

**To sign a single certificate request, use the puppet cert sign command, with the hostname of the certificate as it is displayed in the certificate request.**

**Run the below command to sign a request.**

/opt/puppetlabs/bin/puppetserver ca sign --certname client.mit.com,client.mit.com

*output*

*Successfully signed certificate request for client.mit.com*

*Successfully signed certificate request for agent.mit.com*

**Verify Puppet Agent**

**Once the Puppet master has signed your Puppet Agent certificate, run the following command on Puppet Agent machine to test it:**

sudo /opt/puppetlabs/bin/puppet agent --test

**{NOTE: In case if its show "Error: Could not run: Another puppet instance is already running; exiting" yhis error then you need to stop and restart the puppet agent 2-3 times}**

*output:*

*Info: Using configured environment 'production'*

*Info: Retrieving pluginfacts*

*Info: Retrieving plugin*

*Info: Retrieving locales*

*Info: Caching catalog for client.mit.com*

*Info: Applying configuration version '1567051174'*

**Create manifest**

**Manifest is a data file which contains client configuration’s, written in Puppet’s declarative language or a Ruby DSL. This section covers the basic manifest to create a directory as well as a file on the client machine.**

**Main puppet manifest file is located at /etc/puppetlabs/code/environments/production/manifests directory. Create a new manifest file.**

vi /etc/puppetlabs/code/environments/production/manifests/site.pp

Now add the following lines to the manifest to create a directory on the client machine.

machine 'client.mit.com' { # Applies only to mentioned machine. If nothing mentioned, applies all.

file { '/tmp/puppetdir': # Resource type file

ensure => 'directory', # Create as a diectory

owner => 'root', # Ownership

group => 'root', # Group Name

mode => '0755', # Directory permissions

}

}

**Now, run the following command on the client machine to retrieve the configurations.**

sudo /opt/puppetlabs/bin/puppet agent --test

*Output*

*Info: Using configured environment 'production'*

*Info: Retrieving pluginfacts*

*Info: Retrieving plugin*

*Info: Retrieving locales*

*Info: Caching catalog for client.mit.com*

*Info: Applying configuration version '1567058765'*

*Notice: /Stage[main]/Main/Machine[default]/File[/tmp/puppetdir]/ensure: created*

*Notice: Applied catalog in 0.03 seconds*

**Verify that directory has been created on the client machine.**

ls -ltr /tmp/

drwxr-xr-x 2 root root 4096 Aug 29 06:06 puppetdir

**In Puppet, all the programs which are written using Ruby programming language and saved with an extension of .pp are called manifests. In general terms, all Puppet programs which are built with an intension of creating or managing any target host machine is called a manifest. All the programs written in Puppet follow Puppet coding style.**

**The core of Puppet is the way resources are declared and how these resources are representing their state. In any manifest, the user can have a collection of different kind of resources which are grouped together using class and definition.**

**simple examples for set menifest**

$package = "vim"

package { $package:

ensure => "installed"

}

$packages = ['vim', 'git', 'curl']

package { $packages:

ensure => "installed"

}

**links**

<https://www.itzgeek.com/how-tos/linux/ubuntu-how-tos/how-to-install-puppet-on-ubuntu-16-04.html>

<https://www.bogotobogo.com/DevOps/Puppet/Puppet6-Install-on-Ubuntu18.0.4.php>

<https://www.itzgeek.com/how-tos/linux/centos-how-tos/how-to-install-puppet-4-x-on-centos-7-rhel-7.html> (for centos\_7)

<https://www.tutorialspoint.com/puppet/puppet_manifest_files.htm> (for understand menifest)

<https://puppet.com/docs/puppet/5.3/dirs_manifest.html> (for understand evrything about puppet)