**SEPM**

**Experiment 10**

# Aim:

To install and Configure Pull based Software Configuration Management and provisioning tools using Puppet.

# 

# Theory

Puppet is a Configuration Management tool that is used for deploying, configuring and managing servers.

It performs the following functions:

● Defining distinct configurations for each and every host, and continuously checking and confirming whether the required configuration is in place and is not altered (if altered Puppet will revert back to the required configuration) on the host.

● Dynamic scaling-up and scaling-down of machines.

● Providing control over all your configured machines, so a centralized (master-server or repo-based) change gets propagated to all, automatically. Puppet uses a Master-Slave architecture in which the Master and Slave communicate through a secure encrypted channel with the help of SSL.

Puppet Key Metrics

● Large installed base: Puppet is used by more than 30,000 companies worldwide including Google, Red Hat, Siemens, etc. along with several universities like Stanford and Harvard law school. An average of 22 new organizations per day use Puppet for the first time.

● Large developer base: Puppet is so widely used that lots of people develop for it. Puppet has many contributors to its core source code.

● Long commercial track record: Puppet has been in commercial use since 2005, and has been continually refined and improved. It has been deployed in very large infrastructures (5,000+ machines) and the performance and scalability lessons learned from these projects have contributed to Puppet’s development.

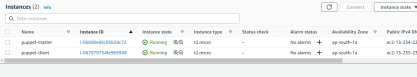
● Documentation: Puppet has a large user-maintained wiki with hundreds of pages of documentation and comprehensive references for both the language and its resource types.

● Platform support: Puppet Server can run on any platform that supports ruby for ex: CentOS, Microsoft Windows Server, Oracle Enterprise Linux, etc. It not only supports the new operating systems but it can also run on relatively old and out-of-date OS and Ruby versions as well.

# Installation:

Prerequisites:

2 Linux Machines on AWS Ec2

Create 2 Ubuntu 20.04 EC2 instances on AWS and take remote access to them using SSH. 

# Implementation

Follow instructions only on the mentioned machine(s)

1. On both master and client, Update Package List.

*sudo apt-get update -y*

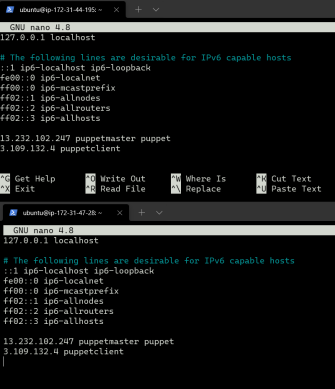
2. On both master and client, set up the hostname resolution.

*sudo nano /etc/hosts*

Now, paste the following lines at the end of each file. These IP addresses are the Public IPv4 addresses of your instances. puppetmaster is your server and puppetclient is the client.

*[puppet master ip] puppetmaster puppet*

*[puppet client ip] puppetclient*



Follow these instructions only on the master machine.

3. Download the latest Puppet Version

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

4. Once the download is complete, install the package by using dpkg

*sudo dpkg -i puppet6-release-focal.deb*

5. Update the package repository:

*sudo apt-get update -y*

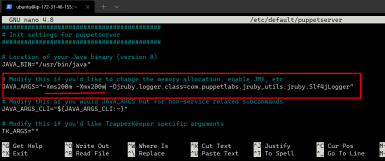
6. Install the puppet server.

*sudo apt-get install puppetserver -y*

Configuration:

7. Open the puppetserver file, under which you need to change the memory size from 2GB to 300MB since we don’t want it to overload our instance which has only 1 GB of RAM.

*sudo nano /etc/default/puppetserver*



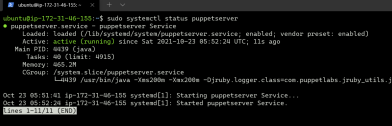
8. Start and enable the Puppet Service

*sudo systemctl restart puppetserver*

*sudo systemctl enable puppetserver*

9. Verify the status of the service.

sudo systemctl status puppetserver



Now that our server is up and running, let’s move to the client machine.

Perform these commands only on the client machine.

10. Download the latest version of Puppet on the client.

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

11. Once the download is complete, use dpkg to install the package.

*sudo dpkg -i puppet6-release-focal.deb*

12. Update the package repository again.

*sudo apt-get update -y*

13. Install the agent using -

*sudo apt-get install puppet-agent -y*

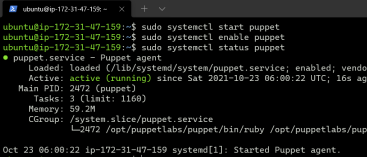
14. Start the puppet service and enable it -

*sudo systemctl start puppet*

*sudo systemctl enable puppet*

15. Check Puppet Service status

*sudo systemctl status puppet*

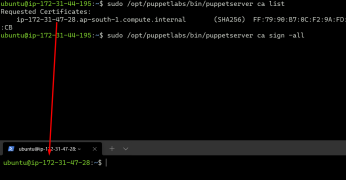


Now that our server and puppet agent both are running, we can proceed to sign the Agent Certificate.

Perform these commands only on the master machine.

16. On the master node, list requested certificates

*sudo /opt/puppetlabs/bin/puppetserver ca list*

**

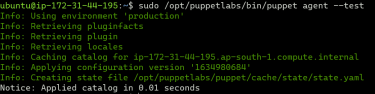
17. Once you spot the requested certificate from the client-side, you can sign it using -

*sudo /opt/puppetlabs/bin/puppetserver ca sign --all*

**

18. To test the connection, you can use the following command

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



If all went good, you’ll get the notice of applying catalog, which means that Puppet was configured properly on both machines.