Deep Chavan
T11-15

LAB ASSIGNMENT 10

AIM: To study Puppet tools in Devops.

LAB OUTCOME:
LO1, LO6 Mapped.

THEORY:

WHAT IS A PUPPET TOOL?
Puppet is an open-source configuration management and automation tool used for deploying, configuring, and managing servers and infrastructure as code. It helps system administrators and DevOps teams automate repetitive tasks, enforce consistent configurations, and ensure that infrastructure is in the desired state, reducing manual effort and minimizing errors.

Features of Puppet:

- **Automation**: Puppet automates the configuration and management of servers and infrastructure, reducing manual tasks and errors.
- **Declarative Language**: It uses a declarative language to specify desired infrastructure states, making it easy to define what the system should look like.
- **Infrastructure as Code (IaC)**: Puppet treats infrastructure configurations as code, enabling versioning, testing, and collaboration like software code.
- **Resource Abstraction**: Puppet abstracts system resources into manageable "resources," simplifying configuration management.
- **Module-Based**: It organizes configurations into reusable modules, streamlining the management of common software and services across different parts of your infrastructure.

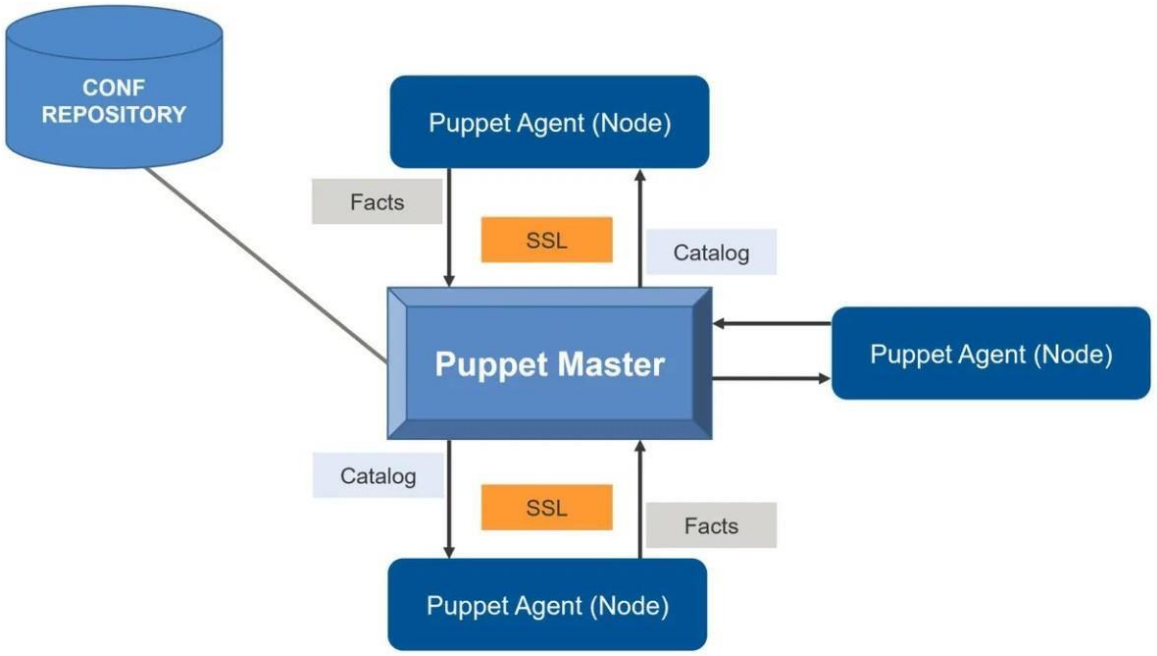
WHAT CAN A PUPPET DO?
The Puppet Server is responsible for:

- Compiling the Catalog File for hosts, based on system, configuration, manifest file, etc. Puppet prepares a Catalog File based on the manifest file, which is a Puppet program used to control the systems running the Puppet Agent. After processing the manifest file, the Puppet Server prepares the Catalog File based on the target platform.
- Sending the Catalog File to Agents when they query the Server.
- Storing information about the entire environment, such as host information, metadata such as authentication keys. ● Gathering reports from each Agent and then preparing the overall report.

HOW DO PUPPETS WORK?
Puppet uses an agent-server model for configuring systems, with agents and a server known as the Puppet Agent and Puppet Server, respectively. The key points are:

- Puppet Agent must be installed on each system for management.
- Agents connect securely to the Puppet Server to receive instructions in a file referred to as the Catalog.

- Agents execute instructions to reach the desired system state and report back to the server.
- Puppet employs a declarative Domain Specific Language (DSL) for configuration attributes, defined in manifests.
- Agents collect facts about themselves and send them to the Puppet Master.
- The Puppet Master compiles catalogs specifying how each node should be configured.
- Catalogs are sent to agents, which then apply the configurations, ensuring consistency. - The Puppet Master manages the entire infrastructure, including compiling catalogs, sending reports, and handling file transfers.
- Communication is secured through SSL/TLS protocols, encrypting traffic between nodes and the master.



Puppet Blocks

Puppet Blocks. Resources- Puppet has many built-in resources like file, user, package and service. The puppet language allows administrators to manage the system resources independently and ensure that the system is in the desired state.

Puppet Resources

Resources are one of the key fundamental units of Puppet used to design and build any particular infrastructure or a machine. They are mainly used for modeling and maintaining system configurations. Puppet has multiple types of resources, which can be used to define the system architecture or the user has the leverage to build and define a new resource.

Puppet Classes

Puppet classes are defined as a collection of resources, which are grouped together in order to get a target node or machine in a desired state. These classes are defined inside Puppet manifest files which are located inside Puppet modules.

Puppet Modules

Modules serve as the basic building blocks of Puppet and are reusable and shareable. Modules contain Puppet classes, defined types, tasks, task plans, functions, resource types and providers, and plug-ins such as custom types or facts. Modules must be installed in the Puppet modulepath.

Puppet Manifest File

A manifest is a file containing Puppet configuration language that describes how resources should be configured. The manifest is the closest thing to what one might consider a Puppet program. It declares resources that define state to be enforced on a node.

PUPPET MANIFEST FILES:

- In puppet, all the programs are written in Ruby programming language and added with an extension of .pp is known as manifests. The full form of .pp is the puppet program.
- Manifest files are puppet programs. This is used to manage the target host system. All the puppet programs follow the puppet coding style. We can use a set of different kinds of resources in any manifest, which is grouped by definition and class.
- Puppet manifest also supports the conditional statement. The default manifest file is available in the /etc/puppet/manifests/site.pp location.

SYNTAX OF A MANIFEST FILE:

A manifest file contains the following components:

1. Resource Declaration
2. Comments
3. Variables
4. Conditional Statements
5. Classes and Modules
6. Include Classes
7. Templates
8. Resource Ordering
9. Node Definitions

EXAMPLE OF MANIFEST FILES:

```
# This is a Puppet manifest file # Define a file
resource file { '/etc/myconfig': ensure => 'file',
owner => 'root', group => 'root', mode     =>
'0644', content => 'This is the configuration
content.',
# Set file content
}
# Define a package resource package
{ 'nginx':
  ensure => 'installed',
}
# Define a service resource service
{ 'nginx':
```

```
  ensure     =>
  'running', enable
             =>
  true, subscribe =>
  File['/etc/myconfig'],
}
#   Node   definition   node
'webserver.example.com'     {
include nginx_config
}
```

WHY DO WE NEED PUPPET MANIFEST FILES?

Puppet manifest files are crucial for:

- Automation: They automate configurations, reducing manual work.
- Consistency: Ensure a uniform state across servers.
- Version Control: Track changes and collaborate using code repositories.
- Modularity: Reuse configurations for efficiency.
- Customization: Adapt configurations using variables.
- Documentation: Clearly document infrastructure setups. ● Scalability: Scale configurations effortlessly.
- Compliance: Enforce security and compliance standards. ● Change Control: Manage changes systematically.

BENEFITS OF PUPPET:

1. Eliminates time consuming, complex and stressful manual configurations of the infrastructure
2. Automates the process of configurations, controlling and managing large numbers (over 100 servers) of servers and other infrastructure
3. Eliminates complex error-prone tasks of automating the infrastructure deployment and configuration
4. It is an inexpensive method of solving the configurations bottlenecks and latency in the speed
5. Puppet is used as a continuous delivery model to the software release cycle by automating the operations and deployment workflow

CONCLUSION:

Through this assignment, I have learnt the concept of Puppet Tools in DevOps, its different components and working. I also understood the fundamentals of Manifest files in Puppet.