PRACTICAL-9

AIM:

Prepare the detailed case study on puppet DevOps tool in cloud business application

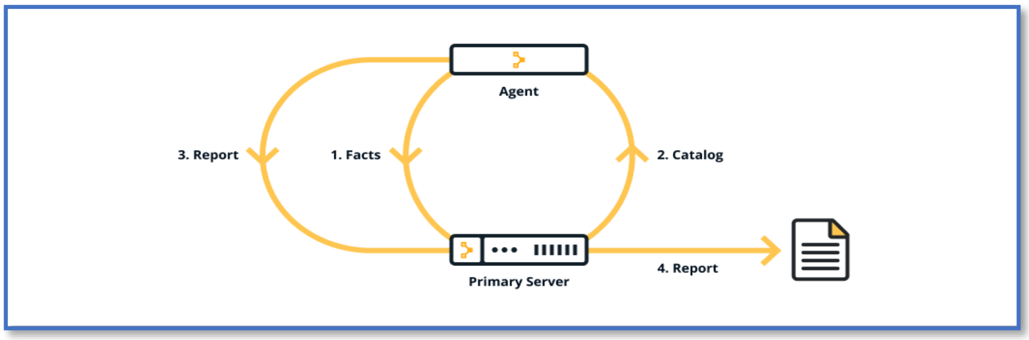
THEORY:

CASE STUDY ON PUPPET DEVOPS

Puppet:

- Puppet is a tool that helps you manage and automate the configuration of servers.
- When you use Puppet, you define the desired state of the systems in your infrastructure that you want to manage.
- You do this by writing infrastructure code in Puppet's Domain-Specific Language (DSL) which you can use with a wide array of devices and operating systems.
- Puppet code is declarative, which means that you describe the desired state of your systems, not the steps needed to get there.
- Puppet then automates the process of getting these systems into that state and keeping them there.

The diagram below shows how the server-agent architecture of a Puppet run works.



Why use Puppet desired state management?

- There are many benefits to implementing a declarative configuration tool like Puppet into your environment — most notably consistency and automation.

- **Consistency**: - Troubleshooting problems with servers is a time-consuming and manually intensive process. Puppet applied the desired state you wanted. You can then assume that state has been applied, helping you to identify why your model failed and what was incomplete, and saving you valuable time in the process.
- **Automation**: - When you manage a set of servers in your infrastructure, you want to keep them in a certain state. If you have 100 or 1,000 servers, a mixed environment, or you have plans to scale your infrastructure in the future, it is difficult to do this manually. This is where Puppet can help you to save your time and money, to scale effectively, and to do so securely.

Key concepts behind Puppet

- Using Puppet is not just about the tool, but also about a different culture and a way of working. The following concepts and practices are key to using and being successful with Puppet.

Infrastructure-as-code

- Puppet is built on the concept of infrastructure-as-code, which is the practice of treating infrastructure as if it were code.
- This concept is the foundation of DevOps, the practice of combining software development and operations. These practices that test code are effectively testing your infrastructure.

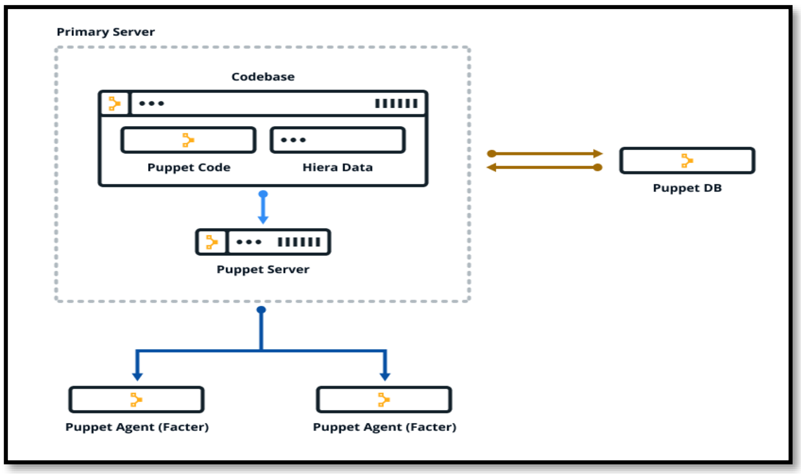
Idempotency

- A key feature of Puppet is idempotency — the ability to repeatedly apply code to guarantee a desired state on a system, with the assurance that you will get the same result every time. Idempotency is what allows Puppet to run continuously.
- It ensures that the state of the infrastructure always matches the desired state. If a system state changes from what you describe, Puppet will bring it back to where it is meant to be.
- It also means that if you make a change to your desired state, your entire infrastructure automatically updates to match.

The Puppet platform

- Puppet is made up of several packages. Together these are called the Puppet platform, which is what you use to manage, store and run your Puppet code.
- These packages include puppetserver, puppetdb, and puppet-agent — which includes Facter and Hiera.
- Puppet is configured in an agent-server architecture, in which a primary node (system) controls configuration information for one or more managed agent nodes. Servers and agents communicate by HTTPS using SSL certificates.
- Puppet includes a built-in certificate authority for managing certificates. Puppet Server performs the role of the primary node and also runs an agent to configure itself.
- Facter, Puppet’s inventory tool, gathers facts about an agent node such as its hostname, IP address, and operating system. The agent sends these facts to the primary server in the form of a special Puppet code file called a manifest.
- Each agent requests and receives its own individual catalog and then enforces that desired state on the node it's running on. In this way, Puppet applies changes all across your infrastructure, ensuring that each node matches the state you defined with your Puppet code. The agent sends a report back to the primary server.
- Hiera, you keep nearly all of your Puppet code, such as manifests, in modules. Each module manages a specific task in your infrastructure, such as installing and configuring a piece of software. Modules contain both code and data.
- The data is what allows you to customize your configuration. Using a tool called Hiera, you can separate the data from the code and place it in a centralized location. This allows you to specify guardrails and define known parameters and variations, so that your code is fully testable and you can validate all the edge cases of your parameters.
- All of the data generated by Puppet (for example facts, catalogs, reports) is stored in the Puppet database (PuppetDB). Storing data in PuppetDB allows Puppet to work faster and provides an API for other applications to access Puppet's collected data. Once PuppetDB is full of your data, it becomes a great tool for infrastructure discovery, compliance reporting, vulnerability assessment, and more. You perform all of these tasks with PuppetDB queries.

The diagram below shows how the Puppet components fit together.



Maintenance cost:

- The cost of software maintenance can be high. However, this doesn't negate the importance of software maintenance. In certain cases, software maintenance can cost up to two-thirds of the entire software process cycle or more than 50% of the SDLC processes.

- The costs involved in software maintenance of VLC Media player is due to multiple factors and vary depending on the specific situation. The older the software, the more maintenance will cost, as technologies (and coding languages) change over time. Revamping an old piece of software to meet today's technology can be an exceptionally expensive process in certain situation.

CONCLUSION:

In this practical, I learnt about the concept and application of puppet Devops.