DEVOPS

What is Puppet

Phone/Whatsapp: +1 (515) 309-7846 (USA)

Email: info@zarantech.com

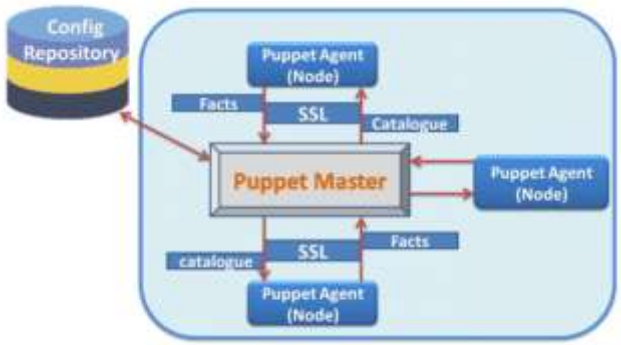
Website: www.zarantech.com

Contents

Why do we use Puppet?

- Expansive Platform Support: Puppet boasts robust platform support, effectively managing systems running Debian/Ubuntu, Microsoft Windows, Red Hat/CentOS/Fedora, and MacOS X.
- Superior Documentation: Puppet sets itself apart with comprehensive and accessible documentation, providing users with valuable resources for utilizing the tool effectively.
- Ongoing Configuration Verification: In contrast to other tools, Puppet continuously verifies deployed configurations at designated intervals, with the flexibility to adjust these intervals as needed.
- Industry Adoption: Major industry players like Google and Red Hat have embraced Puppet, signifying its widespread adoption and reliability.
- Enhanced Productivity: Puppet significantly enhances the efficiency of DevOps professionals and System Administrators, enabling them to work more rapidly and intelligently in their roles.

Understanding the Puppet Architecture and Puppet Components



Puppet relies on a client–server architecture, with the client represented by a Puppet agent and the server by a Puppet master. The key components of the Puppet architecture include:

- **Puppet master:** This component manages all configuration-related information, overseeing deployments and configurations on a designated server.
- **Puppet agent:** Operating on the client side, Puppet agents are the functional machines that are maintained and managed by the Puppet master.
- **Config repository:** This repository stores server-related configurations and nodes, allowing them to be retrieved as needed.
- **Facts:** Global variables containing machine details, such as the operating system and network interfaces, used to analyze the current status of any node.
- **Catalog:** All Puppet configurations are translated into a compiled format known as a catalog, which is then employed on the target system.
- **Manifests:** These files declare all the resources, such as services, packages, or files, that need to be checked and altered, commonly identified by the .pp extension.
- **Modules:** A collection of organized manifest files that facilitate the sharing of resources.
- **Classes:** Similar to programming languages, Puppet employs classes to enhance code organization, readability, and reuse.
- **Resources:** Defined within Puppet codes, resources represent packages, files, users, or commands in the coding block.
- **Nodes:** The servers or clients managed by Puppet agents are referred to as nodes.

How does Puppet work?

Let's break down the Puppet workflow:

- Puppet agents installed on the nodes collect configuration information using facts and send this data to the Puppet master.
- Upon receiving the information, the Puppet master compiles a catalog outlining how the nodes should be configured and sends it back to the agents.
- Agents leverage the catalogs to implement required configuration updates on their nodes and then report back to the Puppet master.
- Additionally, the Puppet master can share these reports with a third-party tool if necessary.

The Connection Between Puppet Master Server and Puppet Agent Nodes

The communication between a Puppet master and a Puppet agent occurs over HTTPS (HyperText Transfer Protocol Secure) with client verification. The Puppet master furnishes an HTTP interface, and when the Puppet agent needs to make a request or submission, it simply sends an HTTPS request to one of the available endpoints within the provided HTTP interface by the Puppet master.

How to Install Puppet on Linux?

Pre-installation Tasks for Puppet

Before proceeding with the installation, we need to address several pre-installation tasks:

1. **Determine the Deployment Type:**

- **Master/Agent Puppet:** This common deployment involves a central Puppet master server hosting all configuration data, with Puppet agent servers on other nodes pulling configurations from the master. When opting for this deployment, the choice of the Puppet master server and system requirements should be decided before proceeding to install Puppet on agent nodes.
- **Standalone:** In this type, all nodes utilize the puppet apply command to compile and apply configurations. Unlike the master/agent Puppet, there is no central Puppet master or Puppet slave. This method helps distribute and reduce the load of compilation compared to the master/agent deployment. However, it may encounter issues with central reporting and external data sources.

2. Verify OS Version and System Requirements:

- For systems running on OS versions with official packages, the installation process is simplified.
- If the system operates on an unsupported OS, Puppet may still run if the Ruby version used is supported, albeit with a more complex installation path.

3. Assess Network Configurations:

For master/agent Puppet deployments, it is imperative to verify the following:

- Firewalls: Ensure the Puppet master server can handle incoming connections on Port 8140, and agent nodes are able to connect to the master on the same port.
- Hostname: Each node should have a unique hostname.
- DNS Configuration: Proper configuration of both forward and reverse DNS is essential.
- Note: The default hostname of the Puppet master is Puppet.

4. **Timekeeping Validation:**

- The system time of the Puppet master must be accurately set. Failure to do so may result in certificates being issued from the past, leading them to be treated as

THANK YOU

Corporate Training Course Catalog

https://lnkd.in/g38XTcqU

Salesforce Learner Community

https://www.linkedin.com/showcase/cloud-technology-learner-community

Get any Salesforce Video Training

https://zarantech.teachable.com/courses/category/cloud-computing

Phone/Whatsapp: +1 (515) 309-7846

Email: info@zarantech.com

www.zarantech.com