## Configuration Management:

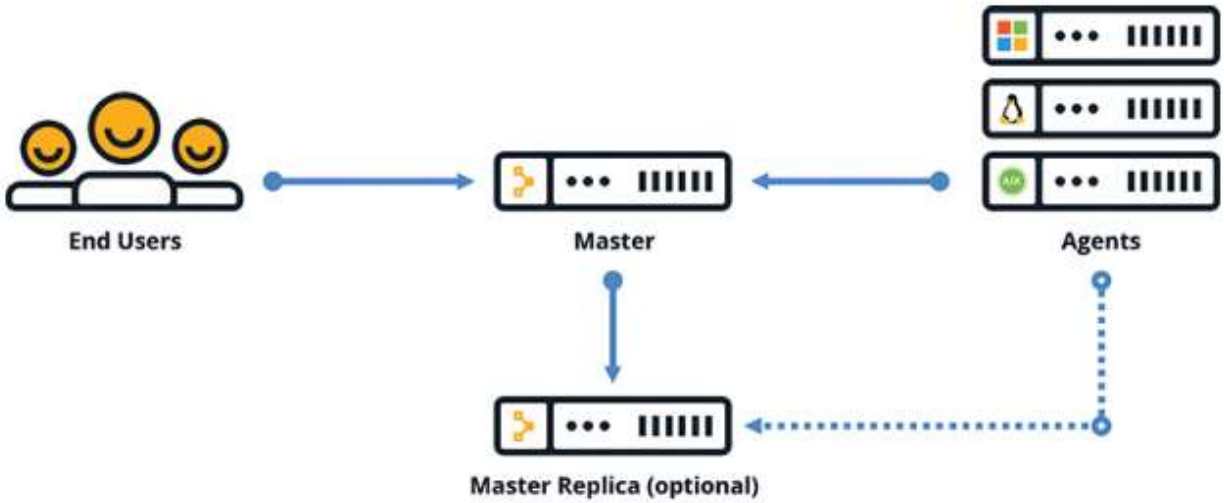
- Configuration management tools offer an automated method to implement.
- Configuration management tools offer an automated method to monitor changes.
- These systems can include the servers, storage, networking devices, and software.
- The main goal is to maintain these systems in known and the determined states.
- Configuration management is important because it allows us to scale infrastructure.
- And software without having to scale staff to manage our network systems & device.
- The Chef, Puppet, Ansible, and the SaltStack are all configuration management tools.
- Which means they are designed to install and manage software on existing servers.
- The Chef, Puppet, Ansible, and the SaltStack are all Configuration Management tools.
- Means Configuration Management are designed to deploy, configure & manage servers.
- These tools are simple to use yet powerful enough to automate IT application settings.


### Agent & Agentless:

- Agent based tools require the installation of an agent on the system you want to manage.
- Agentless tools don't require an agent or software on the system you want to manage.
- Puppet and Chef are two examples of agent-based tools and Ansible is an agentless tool.
- Ansible is agentless, so no software needs to be installed on the client machines or PC.
- Configuration Management Agent-based configuration management is "pull-based".
- Agent on managed device periodically connects with master for its configuration info.
- Changes are done on the master and pulled down and executed by the device etc.
- Configuration Management, Agentless configuration management is "push-based."
- Configuration Management, Agentless a configuration script is run on the master.
- Agentless, the master connects to the device and executes the tasks in the script.

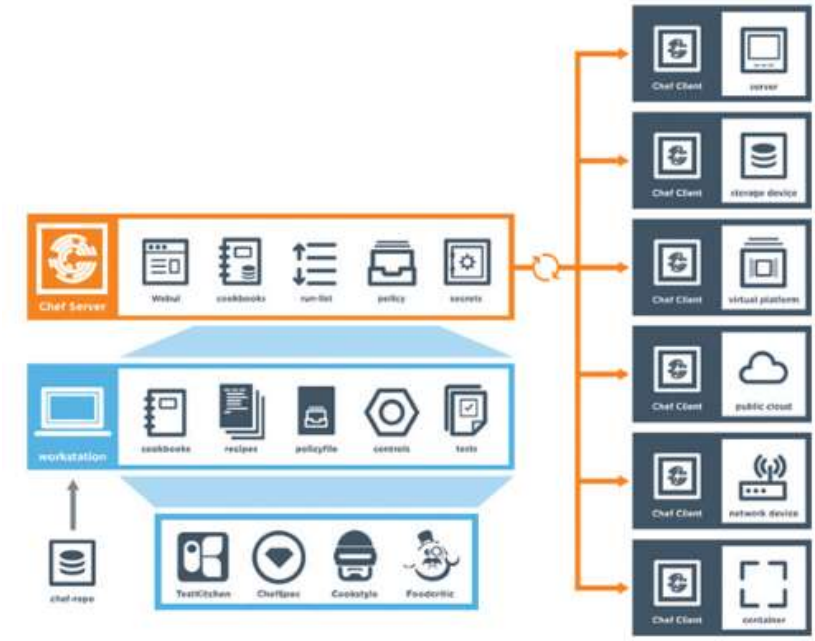
## Puppet:

- Puppet is configuration management tool used for deploying, configuring & managing.
- It is configuration management tool uses for deploying & use master-slave architecture.
- There is a Puppet master on the clients we run Puppet agent that pulls the configuration.
- In these management tool Puppet is one of the oldest and most trusted management tools.
- Puppet provides open-source edition software that can be installed on a wide variety OS.
- Puppet is one of tools in this list which uses its own unique language that is based on Ruby.
- Puppet calls the configuration files manifests; Manifests use own configuration language.
- When it comes to ease of access, the Puppet is a good alternative to Chef and the Ansible.
- Cisco supports the use of Puppet on a variety of devices, such as Catalyst switches, Nexus.
- Cisco supports the use of Puppet on a variety of devices, such as the UCS server platform.
- Puppet works with many different vendors and is one of the more commonly used tools.
- Puppet allows for management & configuration of multiple device types at the same time.
- Puppet agents communicate to the puppet master by using different TCP connections.



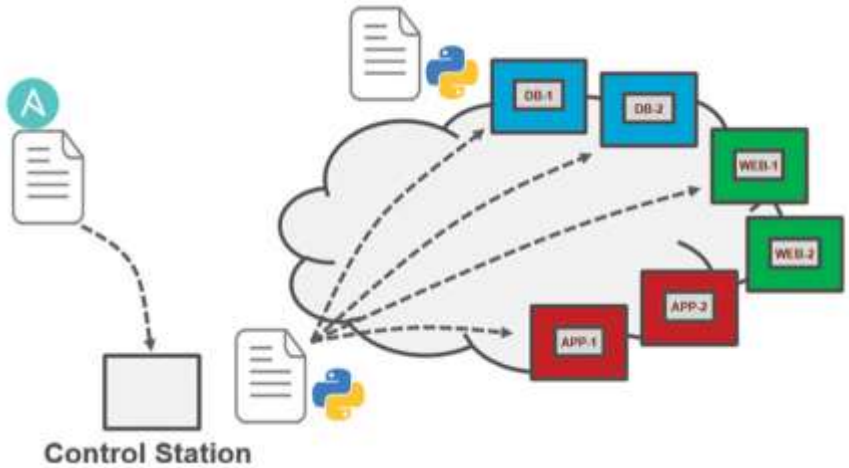

## Chef:

- Chef is an automation platform that configures and manages your infrastructure Network.
- Chef is an automation platform which uses a master-slave architecture, similar to Puppet.
- Chef is a popular, open-source, IaC (Infrastructure as Code) management tool like Puppet.
- It eases administration, configuration & deployment of server resources across a network.
- The Chef server manages nodes and stores the configurations for the nodes.
- Each node runs the chef clients and pulls configuration tasks from the Chef server.
- Chef calls configurations “cookbooks”, Users create, test, and maintain configurations.
- We push these to the Chef server; the Chef uses Ruby DSL for configuration files.
- Configuration management tools function in two different types of models: push and pull.
- Chef’s structure, terminology, and core components are different from those of Puppet.



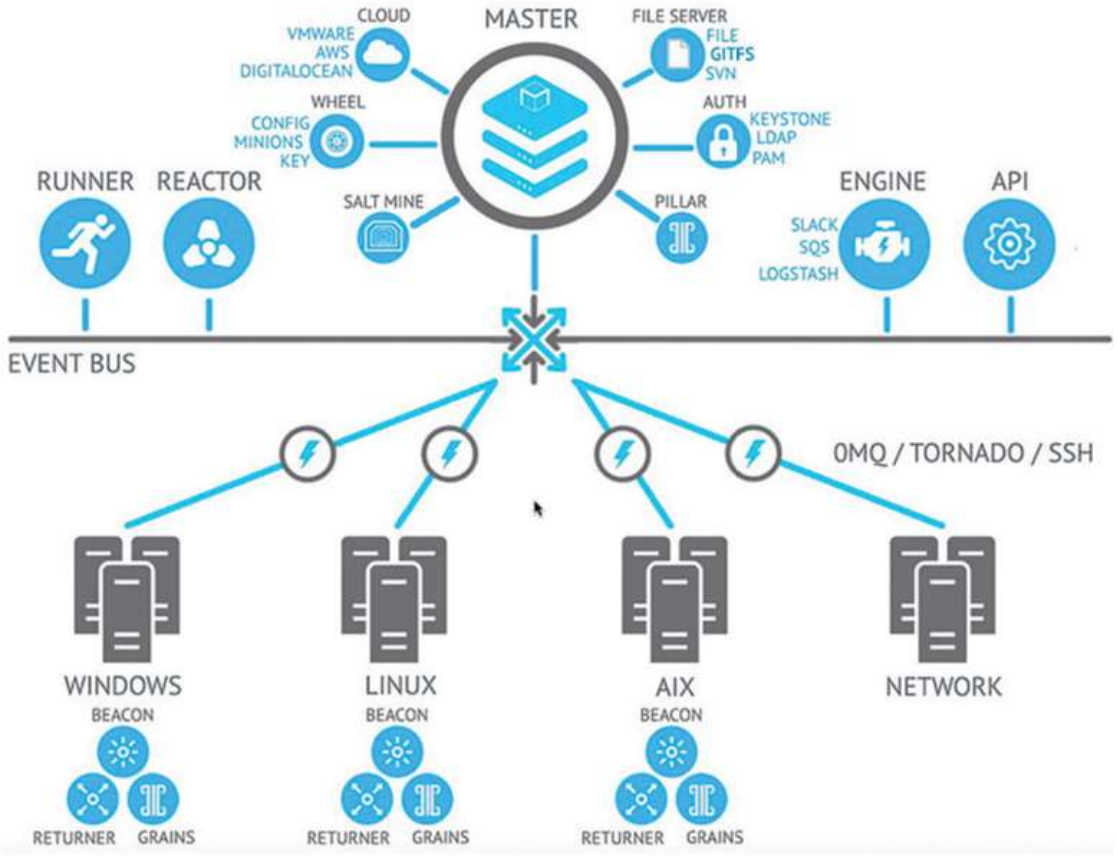

## Ansible:

- Ansible is a configuration and orchestration tool, written in Python which uses YAML.
- We call these tasks playbooks, Ansible is agentless and uses SSH to connect to nodes.
- It sends programs called Ansible modules, runs them, and removes them when finished.
- Ansible is great way to get started with network automation, Playbooks are easy to read.
- It's agentless, you don't have the hassle of installing a server and agents on your nodes.
- It is simple-to-use configuration management tool that provides powerful automation.
- It is more of a provisioning and deployment tool and Primarily useful to IT professionals.
- Ansible helps with application deployment, cloud provisioning, configuration management.
- Ansible virtually do everything a systems administrator does on a daily or weekly basis etc.



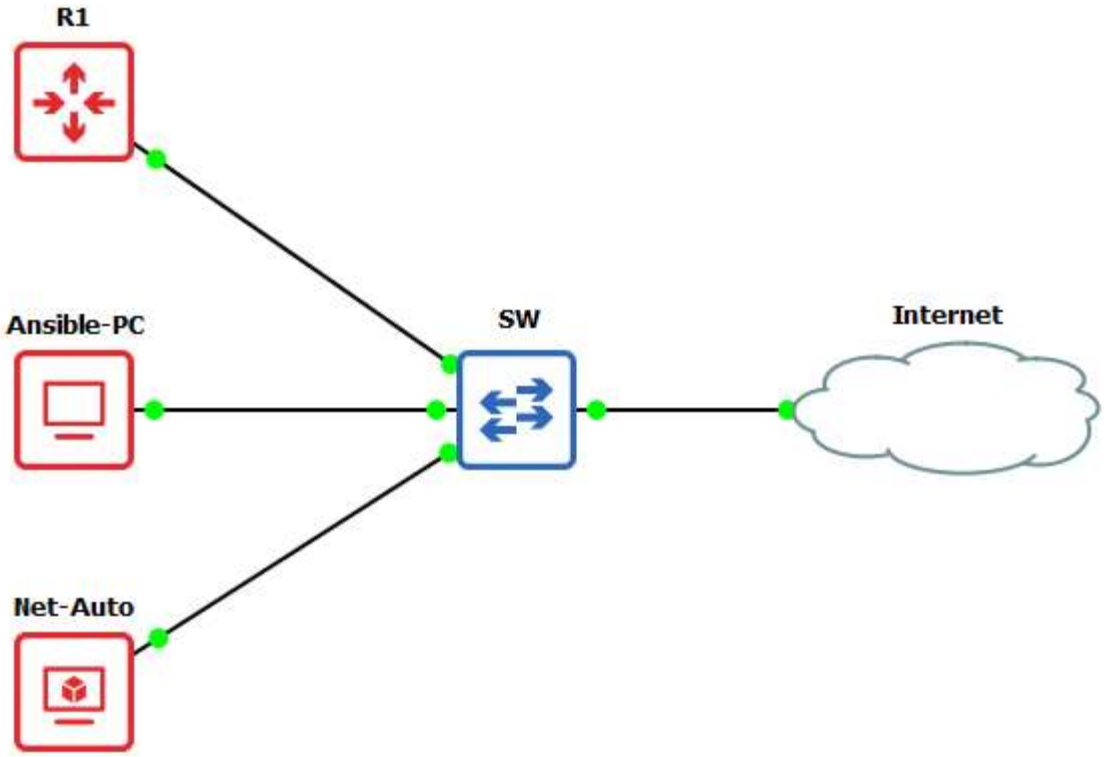
## SaltStack:

- It stands out in this comparison in that it is powerful & robust remote execution tool.
- It is an open-source configuration management software and remote execution engine.
- Like Ansible, SaltStack also uses administrator-oriented YAML, making it easy to learn.
- Written in Python SaltStack configuration management is language agnostic and simple.
- Salt platform uses the push model for executing commands via the SSH protocol.
- SaltStack is the company that created and maintains the Salt Open project and develops.
- SaltStack is open-source software for event-driven IT automation, remote task execution.
- SaltStack, also known as Salt, is a configuration management and orchestration tool.




| Characteristic | Ansible | Chef | Puppet | SaltStack |
|---|---|---|---|---|
| Programming language | Python + YAML | Ruby | Ruby | Python |
| Agent-based or agentless | Agentless | Agent-based | Supports both | Supports both |
| Devices managed | Any device can be “controller” | Chef Master | Puppet Master | Salt Master |
| Tool | Playbook | Cookbook | Manifest | Pillar |

## Ansible Lab:




In network Automation Docker in GNS3 Python and Ansible is already install, lets install in Ubuntu Docker in GNS3 below commands require.

| Install Ansible in Ubuntu Docker |
|---|
| root@Ansible-PC:~# apt update |
| root@Ansible-PC:~# apt install software-properties-common |
| root@Ansible-PC:~# apt-add-repository --yes --update ppa:ansible/ansible |
| root@Ansible-PC:~# apt install ansible |
| root@Ansible-PC:~# ansible --version |

| R1 Configuration |
|---|
| R1(config)#interface e0/0 |
| R1(config-if)#ip add dhcp |
| R1(config-if)#no shutdown |
| SSH Configuration in R1 |
| R1(config)#username admin privilege 15 password 123 |
| R1(config)#ip domain-name test |
| R1(config)#crypto  key generate rsa |
| R1(config)#line vty 0 4 |
| R1(config-line)#transport input all |
| R1(config-line)#login local |

| Ansible Configuration |
| --- |
| root@Net-Auto:~#:vi /etc/ansible/ansible.cfg |
| # uncomment this to disable SSH key host checking |
| host_key_checking = False |
| |
| root@Net-Auto:~# ansible all -i 192.168.114.131, -c network_cli -u admin -k -m ios_facts -e ansible_network_os=ios |

| root@Net-Auto:~# vi 1st-ans.yml |
| --- |
| ---<br><br>- name: Network Getting Started First Playbook<br>  connection: network_cli<br>  gather_facts: false<br>  hosts: all<br>  tasks:<br><br>  - name: Get config for IOS devices<br>  ios_facts:<br>  gather_subset: all<br><br>  - name: Display the config<br>  debug:<br>  msg: "The hostname is {{ ansible_net_hostname }} and the OS is {{ ansible_net_version }}" |
| root@Net-Auto:~# ansible-playbook -i 192.168.114.131, -u admin -k -e ansible_network_os=ios 1st-ans.yml |

```
ok: [192.168.114.131]

TASK [Display the config] ******************************************************
ok: [192.168.114.131] => {
    "msg": "The hostname is R1 and the OS is 15.4(1)T"
}

PLAY RECAP *********************************************************************
192.168.114.131            : ok=2    changed=0    unreachable=0    failed=0
```