Caltech | Center for Technology & Management Education

# Post Graduate Program in DevOps

Powered by simplilearn

Caltech | Center for Technology & Management Education

# Configuration Management with Ansible and Terraform

Powerd by simplilearn

# Getting Started with Configuration Management

# A Day in the Life of a DevOps Engineer

You are working as a consultant in an organization that is looking for a systems engineering process for establishing and maintaining consistency of a product's performance, functional, and physical attributes with its requirements, design, and operational information.

The goal is to keep computers in the intended condition.
The team is also looking for a method of managing and providing computer data centers through computer definition files.
Additionally, the team is also looking for an IT engine used to automate application deployment, service orchestration, cloud services, and other IT tools.

To achieve all of the above, along with some additional features, you will be learning a few concepts in this lesson that will help find a solution for the given scenario.

# Learning Objectives

By the end of this lesson, you will be able to:

- State the basics of Configuration Management
- Define Infrastructure as Code and list its importance
- List down the core components of Ansible
- State the working and architecture of Ansible

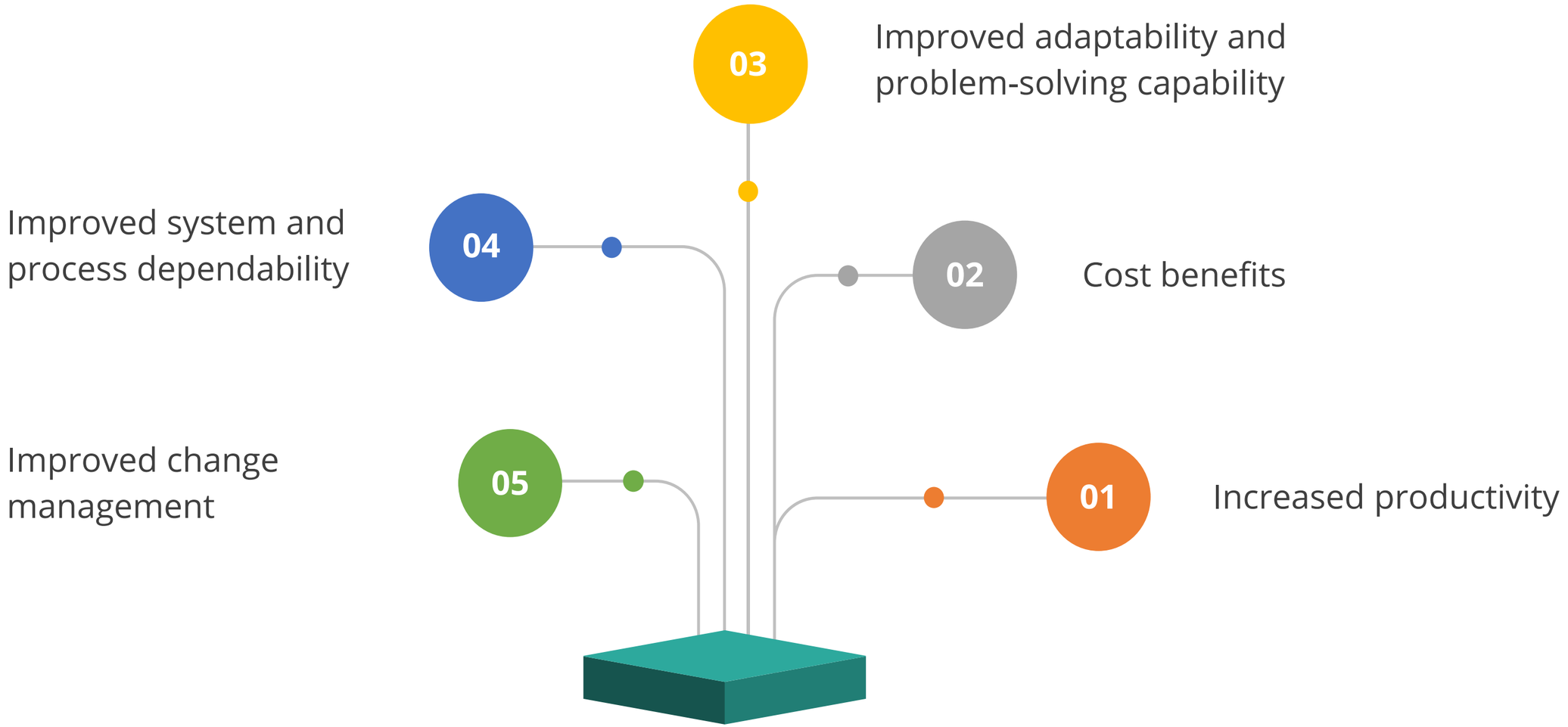
# Understanding Configuration Management (CM)

# Introduction to Configuration Management (CM)

CM is a systems engineering process for establishing and maintaining consistency of a product's performance, functional and physical attributes with its requirements, design, and operational information.

# Benefits of Configuration Management

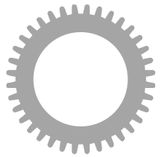
Following are the benefits of Configuration Management:



# Configuration Management Scope

CM tools manage all configuration items in a software.

These configuration items can be software application files, software packages, and software installations.



CM tools cover both software and server configurations.

They also help reduce the time taken to manage configurations on every server.

# Configuration Management Process

Configuration Management Process comprises three operations:

**Configuration Identification**
Identifies the correct configuration that needs to be managed by the CM tool

**Configuration Regulation**
Regulates the way configuration changes are made for the application software

**Configuration Compliance**
Audits and implements compliance on configuration changes made to the application software

# Automating Configuration Management

Configuration Management's objective is to keep computers in the intended condition. Traditionally, system administrators managed this manually or with proprietary programming.

Automation is using software to do operations such as configuration management to cut costs, time, and efforts.

# Features of Configuration Management Tools

Following are the features of CM tools:

01 Manage software configurations effectively

02 Remove manual errors while performing configuration changes for application software

03 Manage configurations in multiple environments

04 Store generic configurations in version control

# Features of Configuration Management Tools

05 Save time by automating generic configuration process

06 Deploy application source code across the infrastructure

07 Eliminate the need for documentation process for performing configuration changes

# Configuration Management Tools

**Puppet**
Ruby DSL-based CM tool used for managing software, systems, and network configuration items

**Chef**
Ruby-based CM tool having integration with most of the cloud-based platforms

**CM Tools**

**Ansible**
Python-based CM tool, also considered as agentless CM tool

**SaltStack**
Python-based open-source CM tool used to remotely manage configuration items

# Understanding Infrastructure as Code

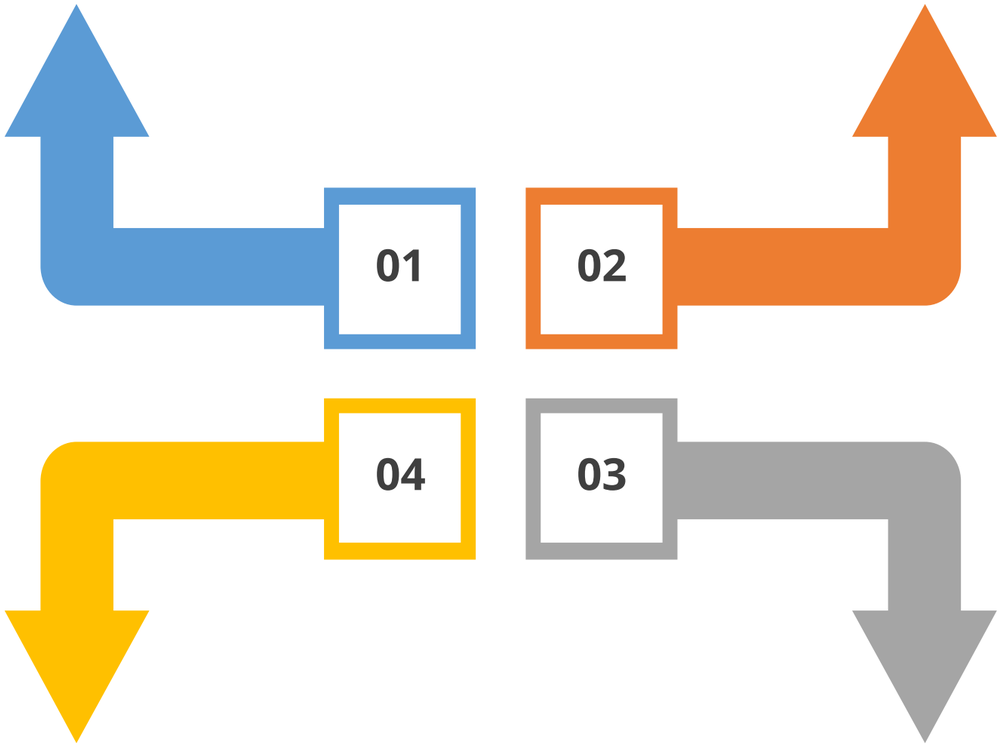
# Infrastructure as Code

Infrastructure as Code (IaC) is the method of managing and providing computer data centers through computer definition files.

# Role of Infrastructure as Code

Following are the roles of IaC:

Manages configuration items




Handles multiple environments with infrastructure scripts or codes

Documents software modifications and infrastructure configurations

Integrates with version control and share with others

# Declarative Vs. Imperative Approaches

## Declarative approach

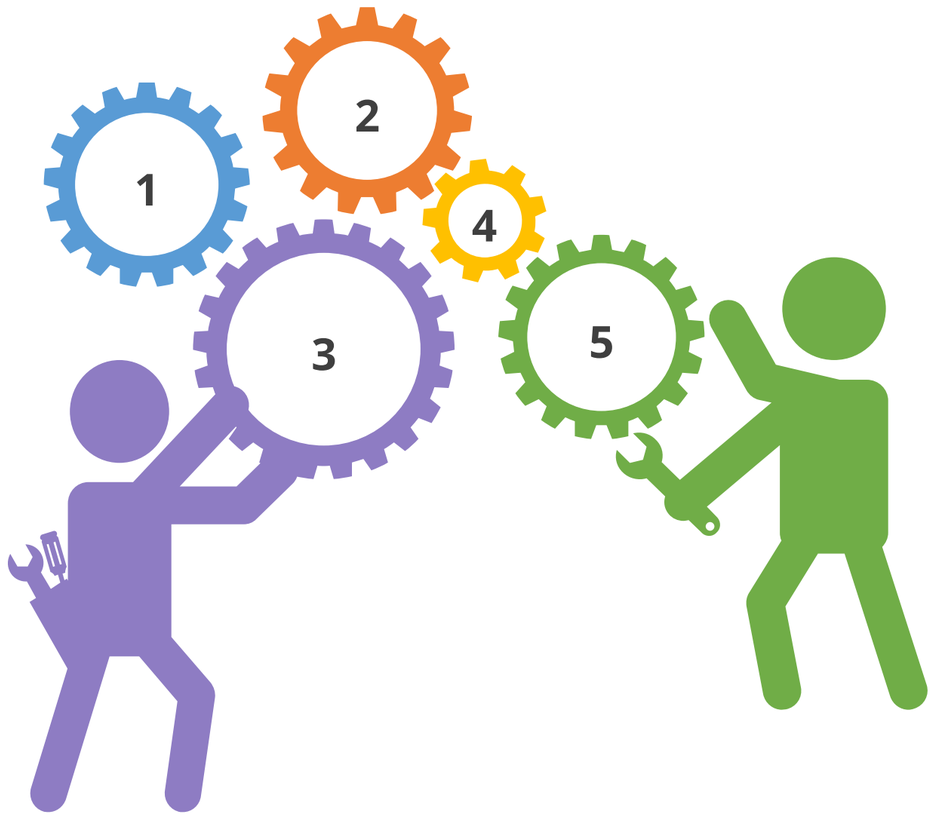
In a Declarative approach, the IaC tool will set up the system for you, which will define the ideal system state, including what services you require and any qualities they must have.

## Imperative approach

An Imperative approach defines the specific commands that must be executed in the precise order to create the intended configuration.

# Benefits of Infrastructure as Code

**IaC approach offers the following benefits:**




- Increase in speed of deployment
- Consistency in configuration
- Risk Minimization
- Increases software development efficiency
- Cost savings

# Introduction to Ansible

# Introduction to Ansible

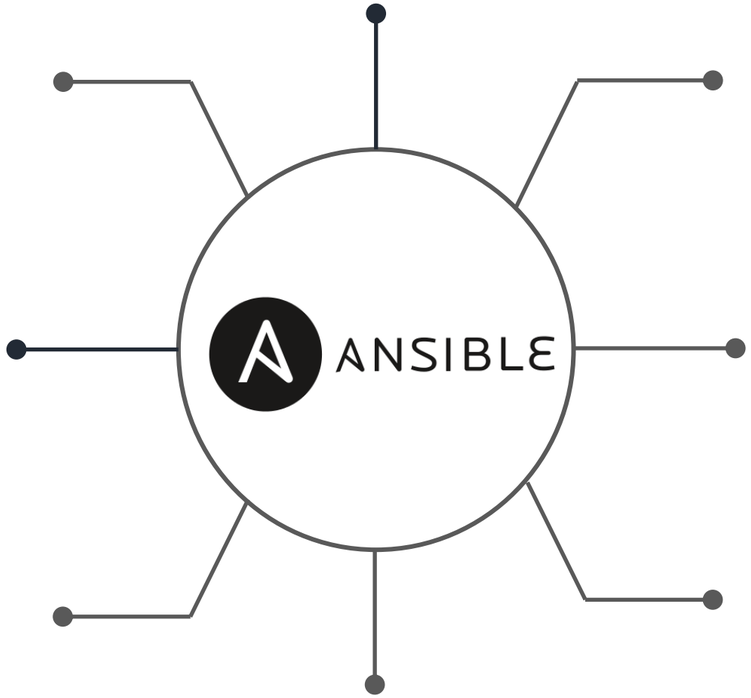
Ansible is an open-source IT engine used to automate application deployment, service orchestration, cloud services, and other IT tools.

# Introduction to Ansible

It is a Python-based, open-source CM tool.

It is an efficient tool.

It can manage both Unix and Windows-based systems.




Ansible controller and nodes are the main components.

It was launched in 2012 and later acquired by Red Hat in 2015.

It can be coded using standard YAML format.

It is an agentless CM tool.

It has an easy learning curve.

# Introduction to Ansible

- Ansible uses Playbook to describe automation jobs that are written in YAML.
- Ansible is designed for multi-tier deployment.
- Ansible models IT infrastructure by interrelating all the systems.

# Introduction to Ansible

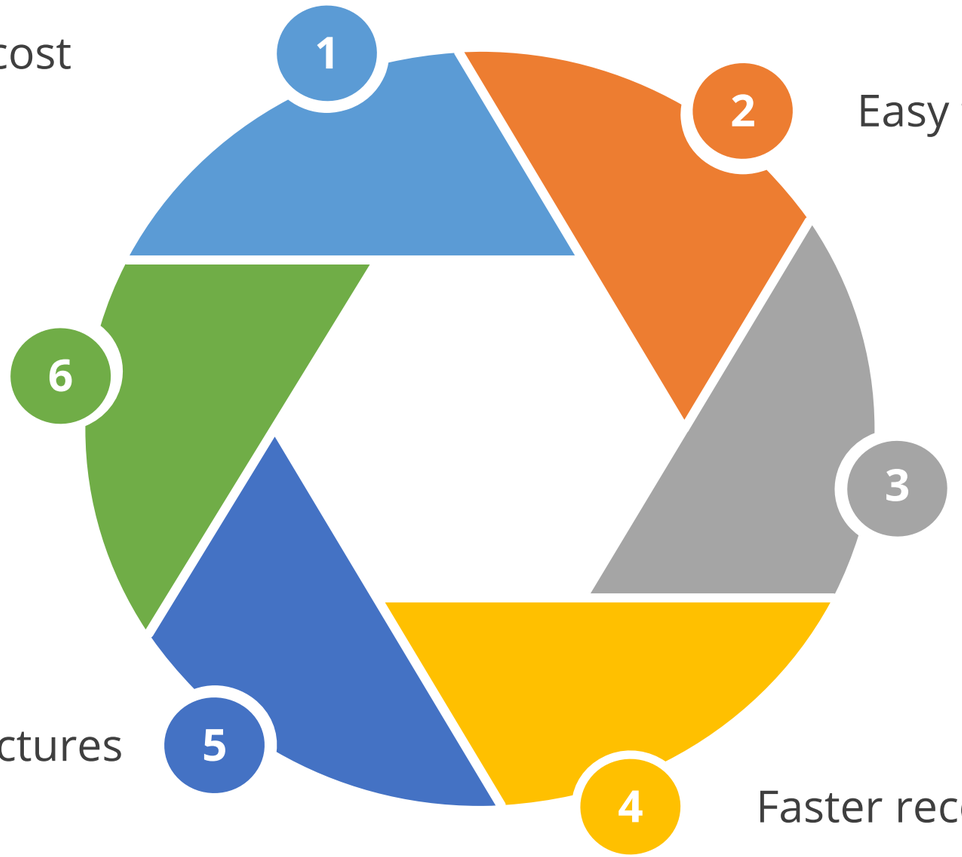
- Ansible is completely agentless and works by connecting nodes primarily through SSH.
- Ansible pushes small programs, called Ansible modules, on the nodes and removes them when finished.
- Ansible manages inventory in simple text files called host files.
- Ansible uses the host files to control the actions on a specific group in the playbooks.

ANSIBLE

# Why Use Ansible?

**Ansible has the following benefits:**

1. Lower system maintenance cost
2. Easy to audit and verify processes
3. Easy to structure release process with distributed and continuous development
4. Faster recovery time
5. Compatible with all infrastructures
6. Easy to audit and verify processes



# Ansible Components

Following are the components of Ansible:

| | |
|---|---|
| **Control Machine** | A machine on which Ansible is installed and acts as a server |
| **Inventory** | Host-file that contains the information about the managed nodes |
| **Playbook** | The entry point of Ansible provisioning written in YAML |
| **Task** | Block of code that defines a single process |

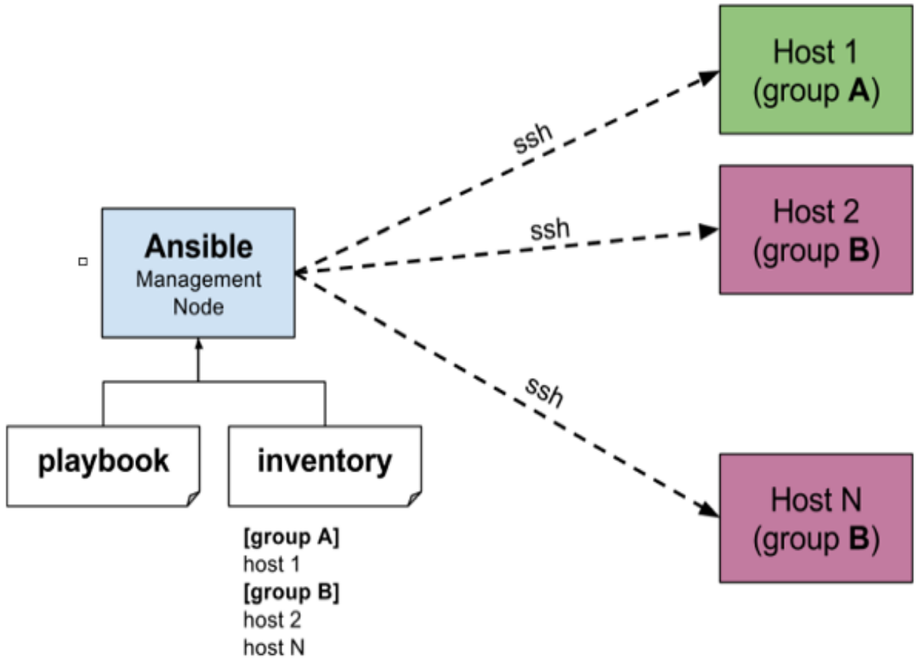
# Ansible Components

Following are the components of Ansible:

| | |
|---|---|
| **Module** | Abstract of a system task like creating and changing files |
| **Role** | A framework that organizes playbooks and other files to facilitate sharing and reuse provisioning |
| **Facts** | Global variables containing information about the system |
| **Handlers** | Tasks that trigger changes in service status |

# Ansible Working and Architecture

Ansible works by connecting to nodes and pushing out small programs called Ansible modules.




Ansible runs modules over SSH by default and removes them when finished.

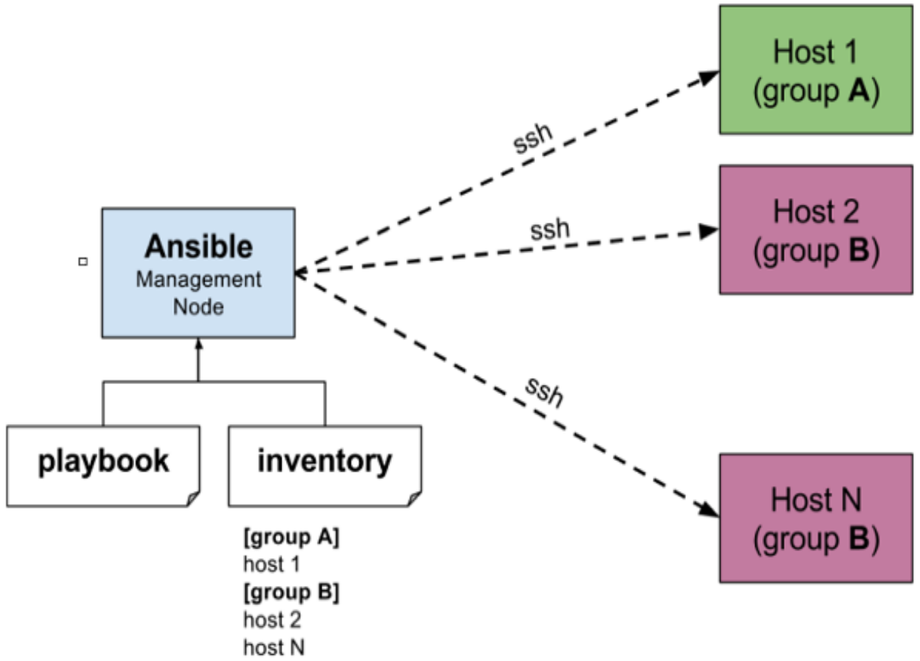
Modules can be stored on any machine without any servers, daemons, or databases.



Source: https://www.tutorialspoint.com/ansible/ansible_introduction.htm

# Ansible Working and Architecture

The management node is the controlling node. It controls the execution of the playbook which is the YAML code written to execute small tasks over the client machines.




The inventory file is the list of hosts where the Ansible modules will run.

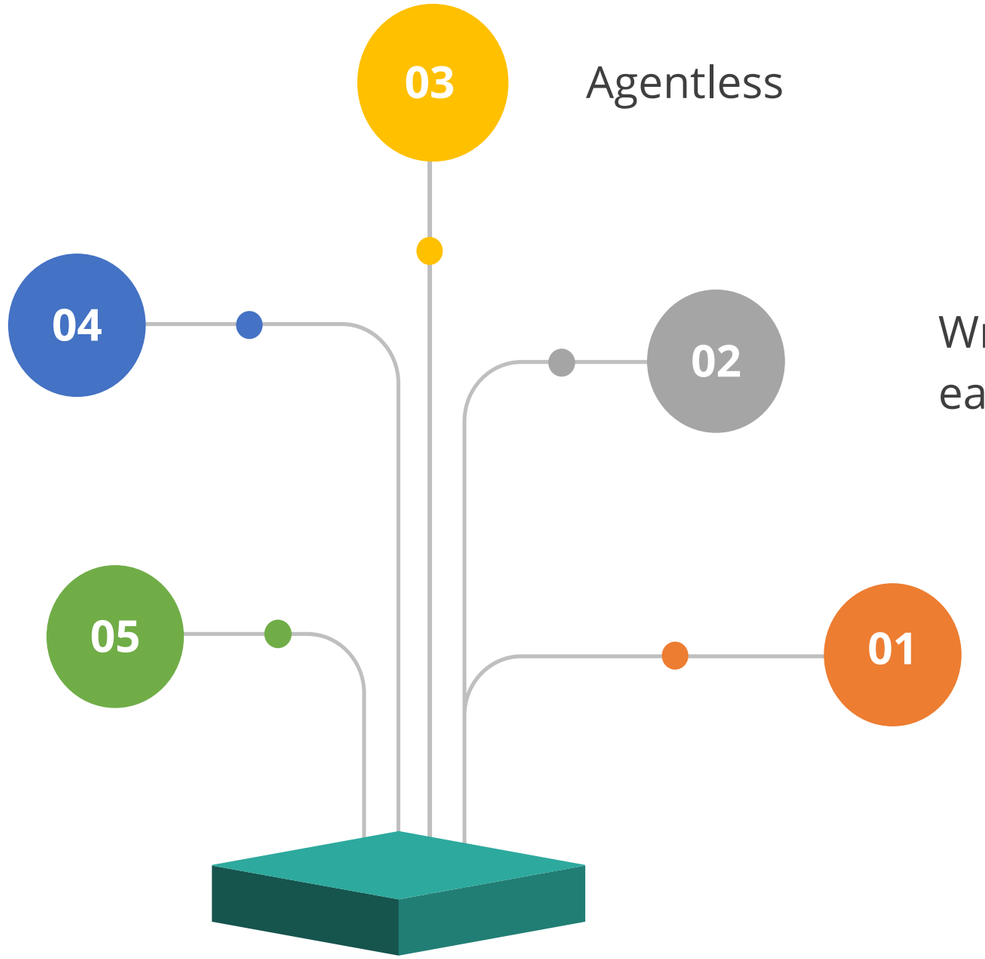
Management node performs an SSH connection and runs modules on the hosts.



Source: https://www.tutorialspoint.com/ansible/ansible_introduction.htm

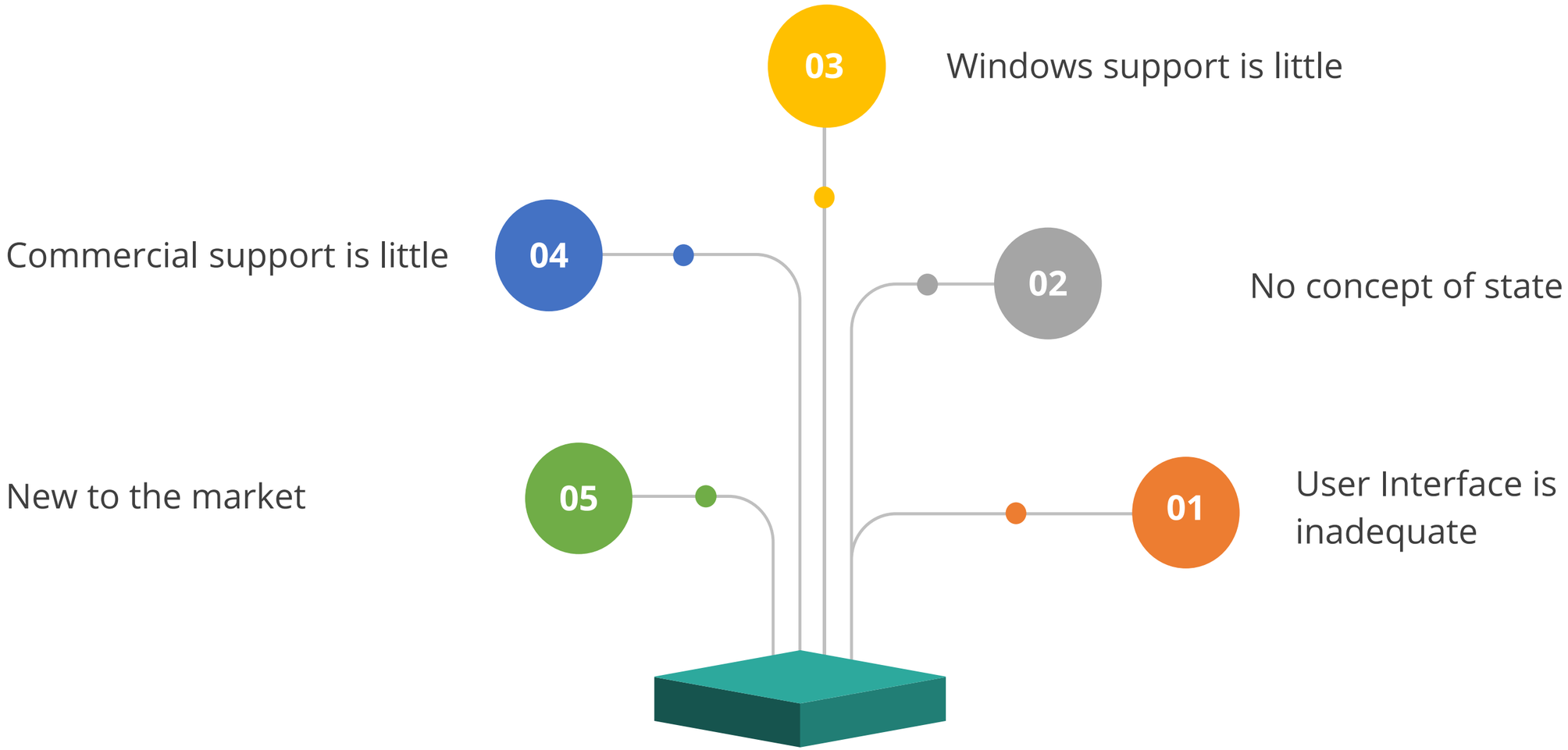
# Advantages of Ansible

Following are the advantages of Ansible:



# Drawbacks of Ansible

Following are the drawbacks of Ansible:



# Ansible Setup

# Assisted Practice

## Setting up Ansible

**Duration: 10 Min.**

**Problem Statement:**

You've been assigned the task of installing and setting up Ansible in your system that will allow to execute the ansible playbooks and configure webservers.

# Assisted Practice: Guidelines

**Steps to be followed:**

1. Installing Ansible
2. Establishing SSH key pair in Linux system
3. Establishing connectivity between Ansible controller and node machine

# Assisted Practice

## Setting up Ansible with Docker

**Duration: 10 Min.**

**Problem Statement:**

You've been assigned the task of installing and setting up Ansible using Docker that will allow you to interact with the various types of infrastructure in your environment by using YAML playbooks.

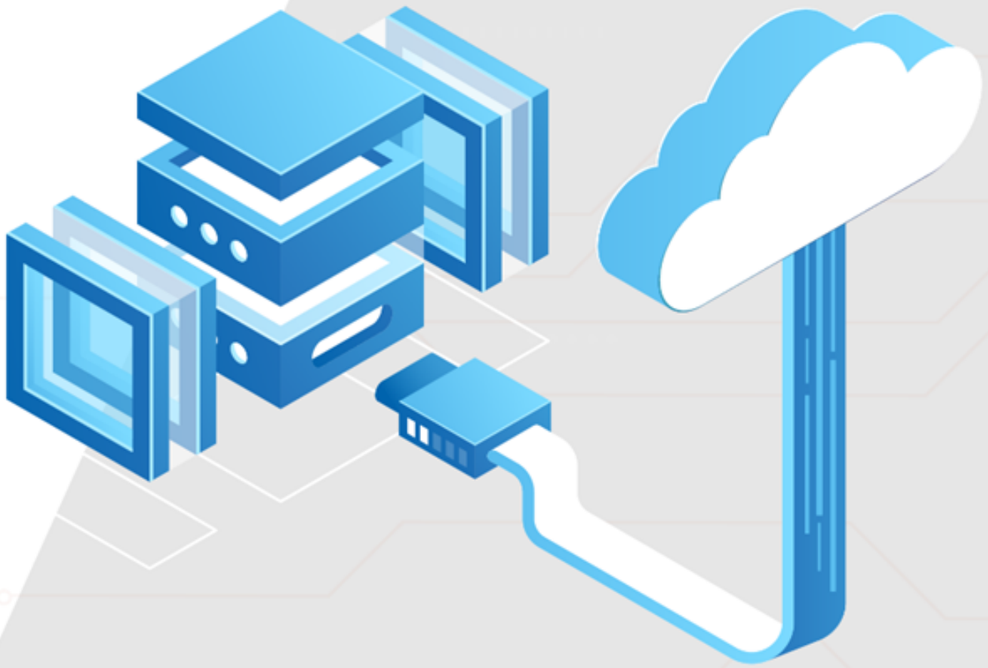
# Assisted Practice: Guidelines

**Steps to be followed:**

1. Installing Docker
2. Creating the Docker file to install Ansible
3. Building the Docker image
4. Running Ansible on Docker

# Key Takeaways

- Configuration management tools manage all configuration items in a software for all environments.
- Ansible Playbooks are written in YAML by default.
- The inventory file is the list of hosts where the Ansible modules will run.
- Each Ansible Playbook works with an inventory file.



# Thank you