What is Ansible?

Ansible is an open source automation platform that is used for configuration management, application deployment task automation etc.

It can also do I-T orchestration to run tasks in sequence and create a chain of events which must happen on several different servers or devices. It is developed in Python language. It is useful while deploying any application using ssh without any downtime.

In simple terms Ansible enables us to define our infrastructure as code(IAC) in a simple declarative manner.

What are the adventages of using Ansible?

* We can easily configure Ansible to run and manage complex processes and functions.
* It is a resource sensitive platform. This means that it optimally utilizes available resources without compromising on the quality of the automated task.
* When Ansible is not managing remote nodes it does not run any background processes. This saves up system memory and does not overwhelm the CPA.
* Most importantly Ansible is powerful reliable and secure as it executes tasks in order and limits the transmission of potentially sensitive data between machines.
* It is easy to set up and configure which is a real advantage when it comes to handling complex processes and large scale deployments with its streamlined efficiency.
* It has proven to be a boon for I.T. administrators as it automates to mundane tasks of automation and system upkeep.
* Ansible has a huge engaging online presence in the form of a support community with thousands of experienced members sharing their knowledge and expertise.
* Ansible does not require any dedicated agents to work on it. It only needs open SSH or python on the remote servers. means the ability to manage remote systems without installing additional agents.
* The core idea in Ansible is that we only run and spools automation as needed and we can repeat it without any side effects.
* Ansible is useful in environments where different users have different levels of access. For example, developers, administrators, and QA engineers have managed access to their respective isolated environments.
* There is no accidental risk of a developer pushing content to production.

Compare Ansible Vs Puppet?

* Ansible is easier to setup than Puppet
* Network Management more easy to using Ansible
* In Terms of scalability, both are highly scalable
* For Ansible 'YAML + Python' and for Puppet 'DSL + PuppetDSL' is used as a configuration language
* Costing of Ansible in production is cheaper than Puppet.
* Ansible is agentless but Puppet needs a customs agent on the remote nodes.

Compare Ansible Vs Chef?

* Ansible is easier to setup than Chef
* Network Management more easy to using Ansible
* In Terms of scalability, both are highly scalable
* For 'Ansible YAML' + Python and for 'DSL + Ruby' is used as a configuration language
* Costing of Ansible in production is cheaper than Puppet.
* Ansible is agentless but Chef needs a customs agent on the remote nodes.

How to set up Ansible?

There are two ways to install it.

1. Using Linux based installation procedure (e.g. using apt or yum)

* For this first add the repository to the APT using command on terminal 'sudo apt-add-repository -y ppa:ansible/ansible'
* Update the repository using command 'sudo apt-get update'
* Now install Ansible using command 'sudo apt-get install -y ansible'

1. Using python installer

* For this first install python on the local machine
* Then using 'sudo pip install ansible' command on terminal install Ansible
* Later on, it can be updated using 'sudo pip install --upgrade ansible' command

Once Ansible is installed, then verify it with command 'ansible --version' on the terminal.

What is the Ansible Roles?

* Roles are a way to group multiple tasks together into one container to do the automation in a very effective manner with a proper directory structure, they are built on the idea of include files and combine them to form clean, reusable abstractions.
* Roles provide a framework for fully independent or interdependent, collections of variables, tasks, files, templates, and modules.
* Each role is typically limited to a particular theme or desired end result, with all the necessary steps to reach that result either within the role itself or in other roles listed as dependencies.
* Roles themselves are not playbooks. There is no way to directly execute a role. Roles have no setting for which host the role will apply to. Top-level playbooks are the glue that binds the hosts from your inventory to roles that should be applied to those hosts.

What is a Ansible Playbooks?

Playbook is a way to send commands to remote computers in a scripted way and has a collection of YAML based files. Instead of using Ansible commands individually to remotely configure computers from the command line, we can configure entire complex environments by passing a script to one or more systems. For example, install and configure something. Basically, a series of resources (e.g. package, copy, file etc) that will achieve our end result in the correct order.

You have to ensure quality of the final deliverables. You have heard about Quality Assurance and Quality Control which are used for managing quality. What is the difference between QA and QC?

Managing the quality of the project deliverables is extremely important. Quality of the deliverables is ensured by doing both Quality Assurance and Quality Control.

Quality Assurance comes from the laid down processes and standards to do the project work. Adherence to the processes and standards assures that the final deliverables will meet the expected standards of quality. This is done proactively through the execution and development.

Quality Control is the process of monitoring the final deliverables and results created by the team after adhering to the chosen processes and standards. Quality Control uses inspection to check if the final deliverable is correct in all respect and meets all the expected specification or not.

What is the Configuration Mangament tool?

Configuration management tool helps to maintain the desired state of a system. It helps to reduce deployment time and helps to keep consistent configuration across similar systems. It greatly reduces the effort required to do repetitive tasks.  It is one of the building blocks of the CI/CD pipeline. Top configuration management tools in the market are Ansible, Chef, Puppet & Salt. Configuration management tool plays a vital role in Infrastructure as code also. Let's take a real-life scenario to understand it more, if 100 web servers need to be built/configured then without Automation or configuration management tool, it can be a very tedious, lengthy and error-prone task. Also, it will require dedicated resource/efforts and there could be chances to have in-consistency in the configuration. A configuration management tool comes here as a savior and can build/configure 100 web servers very quickly with consistent configuration across all servers. In short, Configuration management tool is very helpful in configuring of systems to a particular state with consistent configuration and fewer efforts.

what are Ansible and its uses?

Ansible is a configuration management tool. It helps to automate repetitive tasks and to achieve the desired state of systems. Redhat is an official vendor for Ansible. It is very simple and easy to use and because of this many organizations adopting it rapidly. There is no need to learn any new language to work on Ansible. It uses YAML to write tasks, plays or playbooks. Ansible can be used in two ways, Ansible command line or Ansible Tower. Ansible can complement CI/CD pipeline greatly by making require configuration automatically with consistency. Thus, it helps to speed up the roll-out new changes or products to production. Ansible can be integrated with Jenkins to enhance CI/CD pipeline and to execute configuration tasks automatically from Jenkins. Ansible can also help in defining Infrastructure as a code, so you can deploy as many servers with consistent configuration.

Explain the term "Idempotency"?

For any configuration management tool, Idempotency is a key feature. Idempotency means that you can execute one or more tasks on a server any number of times and it will not make any change if it is already in the desired state.  Idempotency is a key feature of Ansible too. With this, Ansible makes sure that it doesn’t make unnecessary changes on managed hosts. Think of automating a task using scripts and automating the same task using Ansible. If you run a script to perform a task, it will perform required steps on each and every execution of the script, on the other hand, Ansible will make changes only once, even though you execute it a thousand times. Why idempotency is required? So, let's understand it with an example: suppose you need to append a line “this is a test” to a file and you created a script for this. Now, whenever you run this script it will append the same line again and again ( until you are smart enough to include an existence test in the script ). With Ansible, you just write to append line in a file and then Ansible automatically checks, if it is already there or not.

What are the Ansible Role and there purpose?

Role is a group of variables, tasks, files, and handlers that store in the standardized file structure. Roles facilitate reuse and modularization of configuration. For exp: 2 roles have been written, 1 for Apache HTTP and the second one for Apache Tomcat. Now if you need to configure HTTP and Tomcat both on a server than you can just include both roles in the playbook or if you need only one from it then include only 1 role in the playbook. In this way, there is no need for major changes in playbook other than modification of role name, thus provide reusability and modularity.  So, the role is in simple words are small blocks, which can be used as per requirement anywhere to build a big or small wall.

what arethe Ansible Galaxy?

Ansible Galaxy is a hub or community for finding and sharing Ansible content. Roles for provisioning infrastructure, deploying the application and for many other days today, tasks can be found and downloaded from Galaxy. Content on galaxy can be browsed over [https://galaxy.ansible.com](https://galaxy.ansible.com/). Ansible-galaxy command can be used to install, import, list, delete or search roles from/to Galaxy server. One can find playbooks to configure Apache HTTP, Tomcat, Oracle DB or many more as per need and requirement. Instead of writing long playbooks every time, Galaxy can be checked for already created playbooks/roles and if require that playbooks/roles can be modified as per need locally.

What are the Difference Between play and playbook?

Ansible play is a set of tasks that are run on one or more managed hosts. A play may include one or more tasks, and the most common way to execute play is to use a playbook. Play always starts with the top directory, which means it will start from the “- hosts” line always.

Ansible playbook contains one or more plays to have the feasibility of running multiple tasks on a different set of hosts.

Below is the example of a play:

**-** hosts**:** webservers

  tasks**:**

**-** name**:** ensure apache is at the latest version

    yum**:**

      name**:** httpd

      state**:** latest

Below is the example of a playbook:

**---**

**-** hosts**:** webservers

  tasks**:**

**-** name**:** ensure apache is at the latest version

    yum**:**

      name**:** httpd

      state**:** latest

**-** hosts**:** databases

  tasks**:**

**-** name**:** ensure postgresql is at the latest version

    yum**:**

      name**:** postgresql

      state**:** latest

How ansible work?

On the local machine user uses playbook to configure and manage the execution YAML based scripts. Main building blocks of Ansible are playbook, plugins, modules, host inventories, group\_vars, roles, handlers, and tags. Using these components we define our configuration and scripts and that is executed on the remote nodes using SSH tunneling. Commands/steps in scripts are executed in order but execution happens in parallel on multiple remote nodes managed by Ansible. Ansible pushes out small program termed as 'modules'; which are responsible for performing all heavy liftings.

Ansible carries out these modules upon the SSH by default and eliminates them from the remote node machine when finished.

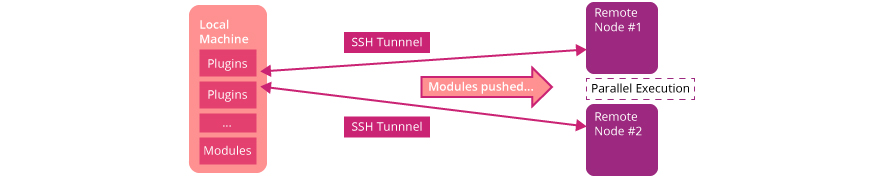
what is the use of ansible?

* It is an open source IT configuration management, deployment, and orchestration tool.
* It is unique from other management tools in many respects, aiming to provide large productivity gains to a wide variety of automation challenges as a more productive drop-in replacement for many core capabilities in other automation solutions.
* To solve major unsolved IT challenges such as clear orchestration of complex multi-tier workflows and cleanly unifying OS configuration and application
* Software deployment under a single solution.
* It is designed to be minimal in nature, consistent, secure, and highly reliable, with an extremely low learning curve for administrators, developers, and IT managers.
* To keep descriptions of IT easy to build, and easy to understand - such that new users can be quickly brought into new IT projects, and longstanding automation content is easily understood even after months of being away from a project.
* To make things powerful for expert users, but equally accessible for all skill levels, ensuring a quicker time to market for IT projects and faster, less-error prone turnaround on IT configuration change.

Ansible workflow?

Workflows are only available with Enterprise-level licenses. Workflows allow configuring a sequence of disparate job templates that may or may not share inventory, playbooks, or permissions. However, workflows have ‘admin’ and ‘execute’ permissions, similar to job templates. A workflow accomplishes the task of tracking the full set of jobs that were part of the release process as a single unit.

A workflow job can have the following states (no Failed



* Waiting
* Running
* Success (finished)
* Cancel
* Error

what is the ansible tower?

Ansible Tower is an enterprise framework for controlling, securing and managing Ansible automation with a UI and restful API.

**It provides the following features**

* Real-time Playbook Output and Exploration
* 'Push Button' Automation( trigger execution with minimum clicks)
* Enhanced and Simplified Role-Based Access Control and Auditing
* Cloud & Autoscaling Flexibility (allows nodes to request configuration on demand)
* The Ideal RESTful API for a systems management application
* Backup and Restore, replicate Ansible tower instance as needed
* Ansible Galaxy Integration ('Don’t Repeat Yourself', using centralized copies of Ansible roles, such as in Ansible Galaxy and uses of requirements.yml file)
* Inventory Support for OpenStack(This allows to easily target any of the virtual machines or images, which are running in the OpenStack cloud)
* Remote Command Execution 12. Integrated Notifications(allows to easily keep track of the status of your automation using e.g. Slack, E-mail, SMS etc)
* Red Hat Satellite 6 and Red Hat CloudForms Integration
* Run-time Job Customization (Bringing the flexibility of the command line to Tower)

Describe Reusability in Ansible?

Ansible playbook files can be reused in multiple ways. Imports and Include are helpful to create/break up multiple small files of a massive playbook. Roles are primarily used to manage various tasks in a package including handlers, variables, plug-in and other modules. They can also be uploaded and shared by Ansible Galaxy. Ansible Tower automatically fetches the roles that playbook needs from Galaxy, GitHub, or your local source control.

what is a 'Module' in Ansible ?

Ansible works by connecting to the remote nodes over SSH and pushing out small programs, called 'modules' to them. These programs are written to be resource models of the desired state of the system. Ansible then executes these modules and removes them when finished.

Ansible ships with a number of readymade modules (called the 'module library') that can be executed directly on remote hosts or through Playbooks. Users can also write their own modules. These modules can control system resources, like services, packages, or files (anything really), or handle executing system commands.

what mean by 'fact' in Ansible ?

Whenever we run an Ansible playbook, Ansible first gathers information ('facts' an exhaustive list of all the environment details) about each host in the play. Facts can be extremely helpful when we’re running playbooks; we can use gathered information like host IP addresses, CPU type, disk space, operating system information, and network interface information to change when certain tasks are run or to change certain information used in configuration files.

To get a list of every gathered fact available, we can use 'ansible munin -m setup' command with the setup module.

what is a handler in Ansible?

* Handlers are just like regular tasks in a playbook, but they will only be run if a task notifies the handler. If a task that would’ve notified the handlers is skipped due to a when condition or something of the like, the handler will not be run.
* Handlers are automatically loaded from roles/<role\_name>/handlers/main.yaml. These handlers can be referenced by any task within the role, or by any tasks within any other role that lists this role as a dependency.
* Handlers will run once, and only once, after all of the tasks complete in a particular play.
* If the play fails on a particular host (or all hosts) before handlers are notified, the handlers will never be run(but we can use the command line flag --force-handlers when running the playbook to run the handler always)

what is a task in Ansible?

* At a basic level, a task is a way to call an Ansible module. In playbook goal of a play is to map a group of hosts to some well-defined roles, represented by things Ansible calls tasks.
* The task file is the main part of a role. If roles/<role\_name>/tasks/main.yaml exists, all the tasks therein and any other files it includes will be embedded in the play and executed.

Describe variable in Ansible?

There are two types of variables that can be defined in a role:

1. Role variables: loaded from roles/<role\_name>/vars/main.yaml
2. Role defaults: which are loaded from roles/<role\_name>/defaults/main.yaml

* The difference between 'variables' and 'defaults' is the precedence order.
* Role defaults are the lowest order variables. Any other definition of a variable will take precedence over a role default.
* Role defaults can be thought of as placeholders for actual data, a reference of what variables a developer may be
* interested in defining with site-specific values.
* Role variables, on the other hand, have a higher order of precedence. Role variables can be overridden, but generally, they are used when the same data set is referenced more than once within a role. If the data set is to be redefined with site-local values, then the variable should be listed in the role defaults rather than the role variables.

What is the meant by Tags in Ansible ?

* If we have a large playbook, it may become useful to be able to run only a specific part of it rather than running everything in the whole Ansible playbook, Ansible supports a 'tags:' attribute for this.
* When we execute a playbook, we can filter tasks based on 'tags' in two ways:
* On the command line, with the --tags or --skip-tags options
* In Ansible configuration settings, with the TAGS\_RUN and TAGS\_SKIP options
* We can apply the same tag to more than one task and do task inheritance as well.

Expain Ansible tags?

Ansible tags are very useful when the playbook is large. It may become useful to be able to run only a specific part of it rather than running everything in the playbook. For example, a playbook creates 3 users but in some cases, you need only 1 user from it, so here tag is a big savior.

During playbook execution, tasks can be filtered out in two ways:

* On the command line, with –tags or –skip-tags options.
* In Ansible configuration settings, with the TAGS\_RUN and TAGS\_SKIP options.

Skip tags can be used to skip one or more tag, it can be defined in the command line as –skip-tags   ‘tag1,tag2’. On the other hand, --tags helps to execute one or tags associated with the mentioned tag.