**Introduction**

Ansible is a widely used tool to automate the software installation and configuration of tools in the servers maintained to host services. It is basically an open-source tool and also provides a GUI as enterprise version called Ansible Tower.

**Configuration Management:**

In our day to day deployments and environments, we use many servers to host the applications. These servers need different tools and software installed on them.

Each type of server requires different set of tools. For ex. A webserver which hosts a Java based web-application needs Java and Tomcat server installed on the server. This webserver denotes the role of the server. It will be easy to install the tools manually if the number of servers we maintain is less.

Generally, the datacenters will have several number of servers to maintain and manually configuring each of them will take more time and effort.

To reduce the time and effort we can use Configuration Management (CM) tools like Chef, puppet, Ansible, Saltstack, etc. modify/maintain the software environment on a server.

The configuration required to be executed on the sever is scripted in a set of files and executed based on need.

Using this approach, we can achieve following:

-      We reduce time and effort using CM tool.

-      We can store the code in a source code management tool and collaboratively work along with other developers.

-      Servers can be recreated easily.

-      Scaling of resources becomes simple.

-      Reduces the cost of operation.

-      Modification of one version to another can be functioned easily.

-      Grouping of resources into different roles.

**Various CM tools:**

Different CM tools available and widely used in the market are as follows:

-      Chef

-      Puppet

-      Ansible

-      Saltstack

-      CFEngine, etc

The architecture of servers being maintained varies from one CM tool to other. In Chef it is a three tier architecture (Server, Node, Workstation), while puppet follows a Master-slave architecture.

Few of the CM tools provide a GUI to trigger deployments from one place.

**Ansible:**

Ansible is a widely used CM tool in the current world and is a good competitor of Chef and puppet. Ansible provides a GUI called Ansible Tower to work with.

Features of Ansible when compared to other CM tools are:

-      Ansible doesn’t follow Master-slave/Server-node architecture.

-      It follows an ssh based bootstrapping of hosts.

-      Deploying the changes can be done from any server which has valid ssh connection to the end-host machines.

-      As it uses the ssh based authentication, Ansible has no limit on the number hosts that can be managed.

-      Also, the core ansible is open-source whereas tower is Enterprise and needs to be purchased.

-      Hence, Ansible is widely used because of its simple and robust architecture.

**Ansible Tower:**

-      Ansible tower is the user interface for Ansible to trigger deploymensts/changes.

-      It contains following configurations:

o    **Project:** SCM repository where the playbooks are stored.

o    **Inventory:** List of hosts that are managed by Ansible. Hosts can be grouped into hostgroups.

o    **Job template:**  It is the settings which we setup providing the information of which playbook needs to be executed on particular hosts.

o    **Jobs:**Once triggered or scheduled, the job template gets executed and we can see the status of Jobs here.

o    Ansible tower follows below sequence of steps to execute the Job

  SCM Checkout.

  Inventory Sync.

  Execution of Playbook.

**Ansible Coding:**

-      Ansible uses yaml format in its files to denote tasks, data, etc.

-      Playbook and Inventory are the key elements of ansible coding.

**Inventory** is the file which contains the list of hosts we manage and needs to be configured.

**Playbook** is the file in which we specify all the required configuration in form of tasks/roles, etc.

-      Playbook contains the series of configuration that needs to be made on the remote host.

-      We may directly add the tasks in the playbook or call the execution of a role.

**Ansible role** is a group of tasks that are combined together into a separate module which is required to configure a particular process/service.

Ex: java-web-server role

-      An Ansible role is nothing but a folder structure containing the following subfolders:

Templates, files, tasks, vars, meta

**Tasks** is the main directory which consists of various yml files in which the **configuration tasks** are defined.

**main.yml** is the default file which gets executed. We can include other files and call them from here.

**Templates** refers to files whose content can be modified dynamically by substituting the variables. These files often have the extension of ‘. j2’(jijna).

-      Templates files are always used along with variables mentioned in vars or ansible facts.

**Files** is a directory in which we can store the files which needs to copied on the remote host. We generally use this option to copy configuration files, etc.

**Handlers**are files which contain helping functions for the actions mentioned in tasks folder.

-      It also has a default main.yml file

**Vars** is the directory which contains the list of variables which are used throughout the playbook role.

-      Default file is main.yml

-      Few variables are fetched from ansible facts which are gathered by ansible on the host machine. Ex: ansible\_distribution returns the OS distribution of host.

**Ansible Resources:**

-      Ansible Resources are configuration elements which used to write the playbooks.

-      Different resources available in Ansible are files, package, service, etc.

**PLAYBOOKS:**

As stated above, playbooksare set of instructions that tell Ansible how to perform specific actions. The playbook should be with. yml extension

If a tasks fails Ansible will stop executing further tasks on that specific system. Playbooks are written in YAML syntax which is a data serialization format like XML and JSON but much better human readable. A convention of YAML is to start every file with three dashes (—).

A sample playbook looks like this –

$cat os-check.yml

-name : Test

Hosts : Sree-Test

User :unixsa

Tasks :

-name : Checks OS version

Shell : cat /etc/redhat-release

Register : etc\_release

* Debug : Msg=”{{etc\_release.stdout\_lines}}”

Explanation :

Here we are performing a task to check the OS version installed by reading the etc/redhat-release file.The output is saved in the etc\_release variable and its printed out on the console.

A playbook can have multiple tasks,multiple host groups(in the above example its just Sree-Test).We can list all the tasks in the playbook using the below command.

$ansible-playbook 0s-check.yml ---list-tasks

[warning] :Log file at var/log/Ansible.log is not writable and we cannot create it ,aborting

Playbook: os-check.yml

Play #1 (Sree-Test) :Test TAGS[]

Tasks :

Checks OS Version TAGS: []

Debug TAGS[]

Some other scenarios and commands used in Ansible:

**CHECKS OS VERSION**:

$ansible all –inventory-file=Ansible-hosts –a “cat /etc/redhat-release”

**CHEKS SERVICE STATUS:**

$ansible all –inventory-file=Ansible-hosts –a “service syslog status”

**CHECKS MEMORY STATUS**

$ansible all –inventory-file=Ansible-hosts –a “free –mt”

**Checks running process:**

$ansible all –inventory-file=Ansible-hosts –a “top –b n1”

Note :

--Inventory file flag specifies the custom host file and –a specifies the linux commands which we want to pass as perameter.We can run multiple command by using the following syntax;

**CHEKS Service Status and Date:**

$ansible all –inventory-file=Ansible-hosts –a “service syslog status”;”date”

**RUNNING A PLAYBOOK :**

$ansible-playbook –I Ansible-hosts os-check.yml

Ansible webservers –m service –a “name =httpd state=started”

Ansible webservers –m ping

Ansible webservers –m command – a “/reboot –t now

Ad Hoc Commands:

An Ad-hoc command is something that you might type in to do something really quick,but don’t want to save for later.

.Runa single ,one off command.

.Run on a full or partial inventory.

.Run on a single host.

$ansible webgroup –m ping

$ andible webgroup –m command –a “uptime”

**Palybooks Execution:Execution**

$ansible –playbook –i production webserver.yml –k –k

$ansible –playbook –i production webserver.yml –f 10 –k –k

$ansible-playbook –i production webserver.yml –list-hosts –k –k

$ansibel-playbook – I production webserver.yml –check –k –k

**Inventory :Example.**

Ansible works against multiple systems in your infrastrure at the same time.It does this by selecting portions of systems listed in Ansible’s inventory file,which defaults to being saved in the location

/etc/Ansible/hosts.

EX:

Mail.example.com

[webservers]

Foo.example.com

Bar.example.com

[dbservers]

One.example.com

Two.example.com

Three.example.com