ANSIBLE REAL TIME SCENARIOS

CONNECT WITH ME ON

INSTAGRAM/TELEGRAM/TWITTER
-SINGAM4DEVOPS

LINKEDIN – PRAVEEN SINGAMPALLI

What is Ansible?

- Ansible is simply an open-source IT engine that automates application deployment, intra service orchestration, cloud provisioning along with the complex automation to support your project requirements.

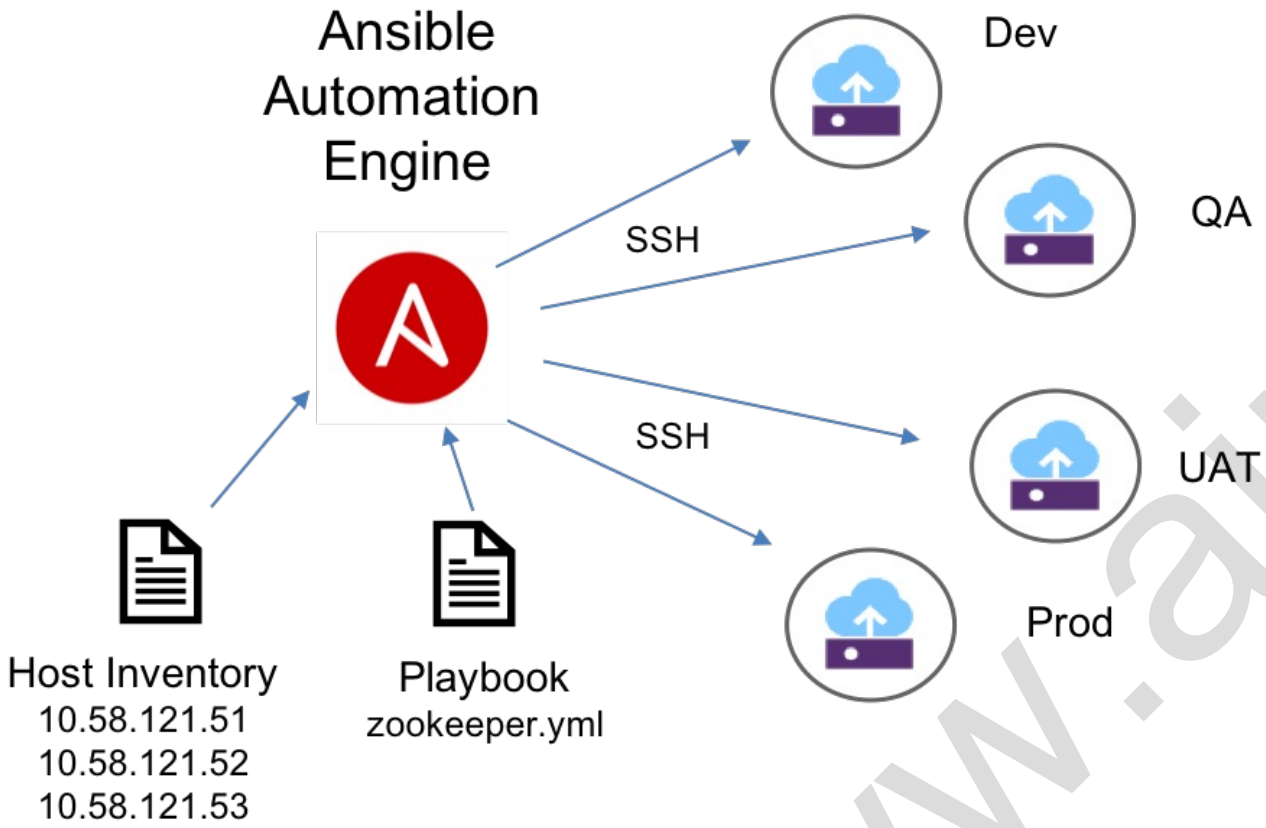
Advantages of Ansible

- **Free**: Ansible is an open-source tool.
- Very simple to set up and use: No special coding skills are necessary to use Ansible's playbooks (more on playbooks later).
- **Powerful**: Ansible lets you model even highly complex IT workflows. 2 Flexible: You can orchestrate the entire application environment no matter where it's deployed. You can also customize it based on your needs.
- **Agentless**: You don't need to install any other software or firewall ports on the client systems you want to automate. You also don't have to set up a separate management structure. Efficient: Because you don't need to install any extra software, there's more room for application resources on your server.

What is Configuration Management?

Configuration management in terms of Ansible means that it maintains the configuration of the product performance by keeping a record and updating detailed information that describes an enterprise's hardware and software.

Ansible Architecture ☺



The management node A is the controlling node (managing node) which controls the entire execution of the playbook. It's the node from which you are running the installation.
The inventory file provides the list of hosts where the Ansible modules need to be run and the management node does an SSH connection and executes the small modules on the host's machine and installs the product/software.

ANSIBLE SETUP

Ansible can be run from any machine with Python 2 (versions 2.6 or 2.7) or Python 3 (versions 3.5 and higher) installed.
Note – Windows does not support a control machine.

Note – Windows does not support a control machine.

INSTALLATION

- 1- Add user on each machine named for example (Ansible)
- 2- Configure SSH login between these servers (control and remotes) without a password
- 3- Install Ansible: [root@ansible-control ~] # yum install -y ansible

ssh root@servera
useradd ansible
passwd ansible
ssh-keygen
cd /etc/suders.d/

Machine − A physical server, VM (virtual machine), or a container.
Target machine − A machine we are about to configure with Ansible.
Task − An action (run this, delete that), etc. managed by Ansible.
Playbook − The YML file where Ansible commands are written and YML is executed on a machine. Ansible.cfg – ansible configuration file 8 Inventory File – a file that contains all the remote ansible nodes

```
cat /etc/ansible/hosts

# we can use '#' for commenting the hostname or ip address in inventory file

# blank lines are ignored by ansible.

# ungrouped hosts are specifying before any group headers like below
192.168.1.1
192.168.1.2

ecanarys.com

[webservers]
# 192.168.1.1
192.168.1.2
192.168.1.3

[databaseservers]
# db1.ecanarys.com
# db2.ecanarys.com
# db3.ecanarys.com
db[1:3].ecanarys.com
db5.ecanarys.com

192.168.1.4
192.168.1.6
```

Ansible - Ad hoc Commands

- **ansible <host-pattern> -m <module-name> -a “<module-command>”**
- Transferring file to many servers/machines
- $ansible abc -m copy -a "src = /etc/yum.conf dest = /tmp/yum.conf”

Creating new directory

- $ ansible abc -m file -a "dest = /path/user1/new mode = 777 owner = user1 group = user1 state = directory”

Transferring file to many servers/machines

- $ Ansible abc -m copy -a "src = /etc/yum.conf dest = /tmp/yum.conf"

The following command checks if yum package is installed or not, but does not update it.

- $ Ansible abc -m yum -a "name = demo-tomcat-1 state = present"

Ansible – Playbooks

- Playbooks are one of the core features of Ansible and tell Ansible what to execute.
- **Ansible Different YAML Tags**
- name
- hosts
- vars
- tasks

Ansible – Roles

Roles provide a framework for fully independent, or interdependent collections of variables, tasks, files, templates, and modules.

In Ansible, the role is the primary mechanism for breaking a playbook into multiple files. This simplifies writing complex playbooks

Each role is basically limited to a particular functionality or desired output

Creating a Role Directory

The above command has created the role directories.

$ mkdir roles
$ cd roles
$ ansible-galaxy role init tomcat

Role tomcat was created successfully

```
$ tree vim
vim
├── defaults
│   └── main.yml
├── files
├── handlers
│   └── main.yml
├── meta
│   └── main.yml
├── README.md
├── tasks
│   └── main.yml
├── templates
├── tests
│   ├── inventory
│   └── test.yml
└── vars
    └── main.yml

8 directories, 8 files
```

Ansible Playbook to Copy A File

```
---
  - name: ply to collect host info
    hosts: servera,serverb,serverc,serverd
    become: true
    user: devops
    tasks:
     - name: collect host info
       copy:
           content: "{{ ansible_hostname }} {{ ansible_processor_count }} {{ansible_default_ipv4.address }} {{
ansible_default_ipv4.macaddress }}"
           dest: /root/hostinfo.txt
```

Handlers and Notifiers

```yaml
---
- name: play to check a package
  hosts: test
  become: true
  user: singam
  ignore_errors: true
  tasks:
    - name: check for package
      yum:
        name: http
        state: present
      register: value

    - name: call a handler
      shell:
        cmd: echo ""
      notify: a
      when: value | failed

    - name: check for service
      service:
        name: http
        state: started
      register: value2

    - name: call b  handler
      shell:
        cmd: echo ""
      notify: b
      when: value2 | failed

    - name: call c  handler
      shell:
        cmd: echo ""
      notify: c
      when: value | failed and value2 | failed
```

```yaml
handlers:
  - name: a
    debug:
      msg: "Installation failed"
  - name: b
    debug:
      msg: "Service failed"
  - name: c
    debug:
      msg: "Playbook was Unsuccessful"
```

Download an artifact and unzip

```
---
- name: play to download the the jar file from jfrog and unarchive
  hosts: devops
   become: true
  user: singam
  tasks:
    - name: create a folder
      file:
        path: /var/deploy
        state: directory
    - name: download the tart
      get_url:
      url: https://artifactory.com/flipcart.zip
      dest: /var/tmp/
  - name: uzipz
     command: unzip -o /var/tmp/flipcart.zip -d /var/deploy
```

ANSIBLE TAGS SCENARIO TO DEPLOY

```
---
- name: play to deploy files in grouped servers
  hosts: all
  become: true
  user: singam
  tasks:
    - name: create a tar file
      command: tar cfz /var/tmp/production.tar.gz /var/www/html
      when: inventory_hostname in groups['production’]
    - name: create a tar file
      command: tar cfz /var/tmp/backup.tar.gz /var/log/httpd
      when: inventory_hostname in groups['backup']
```

Install APACHE and configure the files using ansible modules

```
---
- name: play to install apache
  hosts: devops
  become: true
  user: singam
  tasks:
    - name: Install http package
      yum:
        name: httpd
        state: present
    - name: Download httpd.conf
      get_url:
        url: https://artifact/singam/httpd.conf.j2
        dest: /etc/httpd/conf/httpd.conf
        force: yes
    - name: create index.html
      lineinfile:
        path: /var/www/html/index.html
        line: "Hello from {{ ansible_hostname }}"
        create: yes
    - name: start and enable httpd
      service:
        name: httpd
        state: started
        enabled: true
```

```
echo "hello" > .file1
ansible-vault create crypto.yml --vault-password-file=.file1

---
- password: SINGAM4DEVOPS

ansible-vault view crypto.yml --vault-password-file=.file1

---
- password: SINGAM4DEVOPS


---
- name: ansible-vault
  hosts: dev
  become: true
  user: singam
  vars_files:
    - crypto.yml
  tasks:
    - name: create a folder
      file:
        path: /test/vault
        state: directory

    - name: download the tar
      get_url:
        url: http://artifactory.com/app.zip
        dest: /var/tmp/

    - name: uzip
      command: unzip -o -P {{ password }} /var/tmp/app.zip -d /test/vault
```

ANSIBLE VAULT

ansible-playbook unarchive.yml \ --vault-password-file=.file1

File - unarchive.yml

Ansible Roles

tasks – contains the main list of tasks to be executed by the role.

handlers – contains handlers, which may be used by this role or even anywhere outside this role.

defaults – default variables for the role.

vars – other variables for the role. Vars has the higher priority than defaults.

files – contains files required to transfer or deployed to the target machines via this role.

templates – contains templates which can be deployed via this role.
meta – defines some data / information about this role (author, dependency, versions, examples, etc,..)