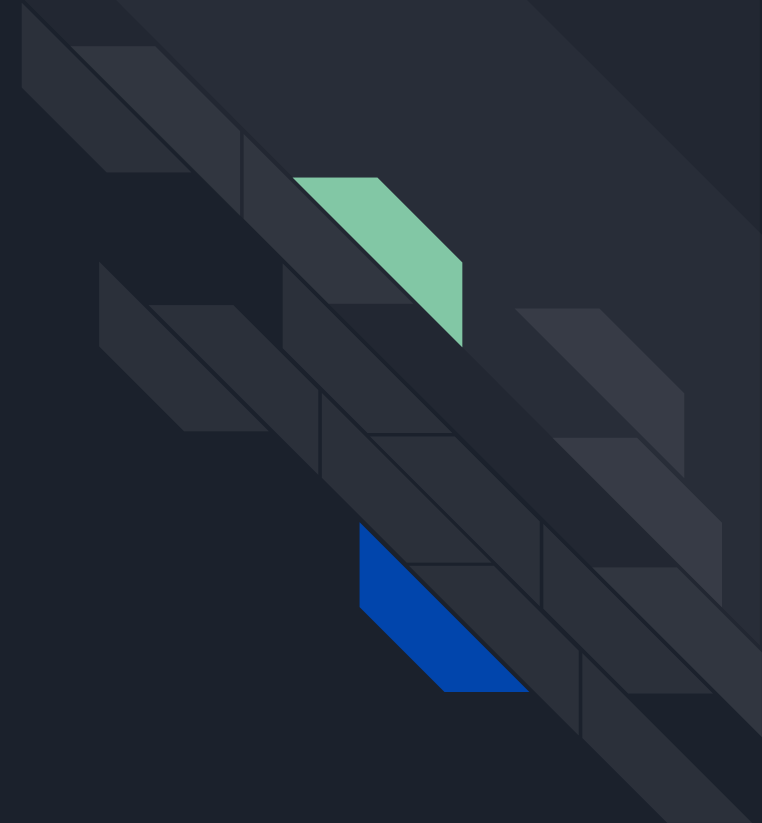
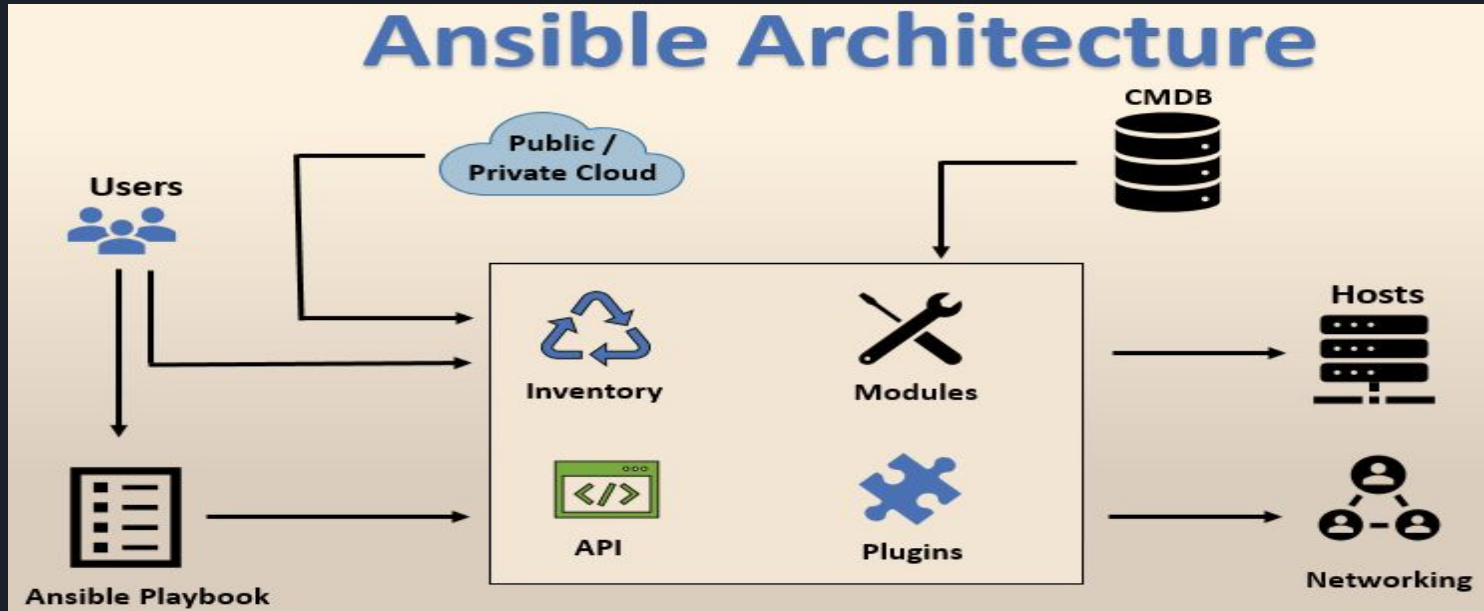


Ansible

1. Ansible – IT Automation Tool
2. Ansible is written in Python, Launched in early 2013.
3. Acquire Ansible by Red hat 2015 – they used Open stack.
4. Playbooks are written in **YAML** format.

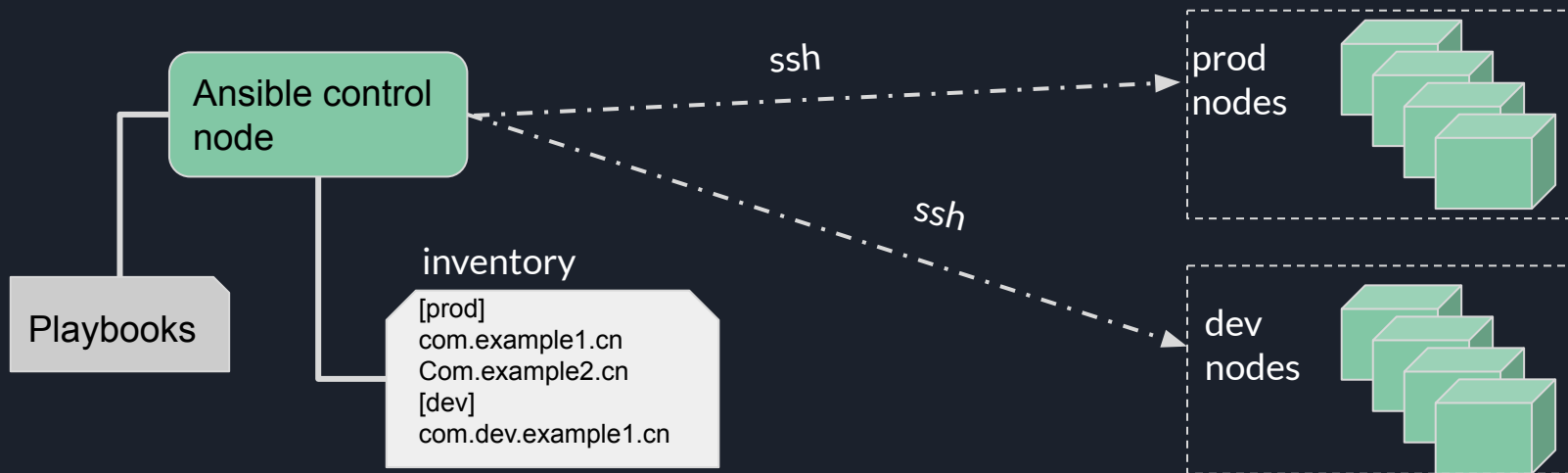


Architecture



What is Ansible? How it helps in automation?

Ansible is an IT automation configuration management tool intended to facilitate the management of remote servers. Ansible requires Python (version 2.7 or 3.5 and higher) to run. Ansible is run from a centralized control node and can manage any server accessible over SSH. Remote servers that are managed by Ansible are called managed nodes.





Why Ansible

- Open Source Software : It is a free open source application
- Agent-less : No Need for agent installation and management
- Declarative not Procedural
- Easy to use / Human Readable
- Python back-end fast process / Ansible is modular. 400+ modules
- Configuration roll-back in case of error.



How to Run Ansible

- **Ad-hoc command:** Ansible inventory -m
- Playbook: Ansible-Playbook
- Roles: Ansible-galaxy
- Ansible Tower: Automation Framework / UI and restful API



Ansible ad-hoc Commands

- Ad-hoc commands are one of the simplest ways of using Ansible.
- These are used when you want to issue some commands on a server or bunch of servers.
- The ad-hoc commands are not stored for future use, but it represents a fast way to interact with the desired servers.



Ad-hoc Commands

- For example, you have to reboot all your company servers. For this, you will run the **Ad-hoc commands** from '/user/bin/Ansible'.
- Command to check the ping status.
- **Ansible all -m ping**



Ansible Playbook

- Ansible ad-hoc commands as **shell commands** and a playbook as a **shell script**.
- It contains a list of tasks (plays) in an order they should get executed against a set of hosts or a single host based on the configuration specified.
- Playbooks are written in **YAML**, in an easy human-readable syntax

Sample Playbook

```
- name: Playbook
  hosts: webservers
  become: yes
  become_user: root
  tasks:
    - name: ensure apache is at the latest version
      yum:
        name: httpd
        state: latest
    - name: ensure apache is running
      service:
        name: httpd
        state: started
```

1 Name of Playbook

2 HostGroup Name

3 Sudo (or) run as different user setting

4

Tasks

Running playbooks

Playbooks are the files where we mention a task or a state in a declarative statements.

For example below we run a example of playbook file for nginx installation.

```
$ ansible-playbook -i /home/ansible/inventory /home/ansible/install-nginx.yml
```

```
--- # install nginx
- hosts: node.kb.liquidweb.com
  become: yes
  tasks:
    name: install epel
    yum:
      name: epel-release
      state: installed

    name: install nginx
    yum:
      name: nginx
      state: installed
```



```
tony@MacBook-Pro ansible % ansible-playbook -i inventory.yml nginx.yml

PLAY [web] *****

TASK [Gathering Facts] *****
ok: [ubuntu]

TASK [apt-get update] *****
ok: [ubuntu]

TASK [install nginx] *****
ok: [ubuntu]

TASK [create geolink www directory] *****
ok: [ubuntu]

TASK [delete default nginx site] *****
ok: [ubuntu]

TASK [copy nginx site.conf] *****
changed: [ubuntu]

RUNNING HANDLER [restart nginx] *****
changed: [ubuntu]

PLAY RECAP *****
ubuntu : ok=7  changed=2  unreachable=0  failed=0  skipped=0  rescued=0  ignored=0
```

Difference b/n AD HOC and Playbook

AD HOC command

`ansible webserver -m yum -a "name=httpd state=latest"`

Ansible Playbook


```
---  
- name: playbook name  
  hosts: webserver  
  tasks:  
    - name: name of the task  
      yum:  
        name: httpd  
        state: latest
```




Ansible Galaxy


- Ansible Galaxy is a galaxy website where users can share roles and to a command-line tool for installing, creating, and managing roles.
- Lots of people share roles in the Ansible Galaxy.
- Ansible roles consist of many playbooks, which is a way to group multiple tasks into one container to do the automation in a very effective manner with clean, directory structures.


Galaxy Dashboard


 GALAXY


AboutHelpDocumentationLogin


 Home


 Search


 Community


 Please take our [Ansible GalaxyNG Roadmap Survey](#) to help us port Galaxy to the new [galaxy_ng](#) codebase.


 Home


 **Most Popular**


 System


 Development


 Networking


 Cloud


 Database


 Monitoring


 Packaging


 Playbook Bundles

 Security

 Web

 **Download**

 **Share**

 **Featured**



Ansible Tower

- **Ansible Tower** is a web-based solution that makes Ansible even more easy to use for IT teams of all kinds.
- Provides role-based access control, job scheduling, integrated notification, and graphical inventory management
- It's designed to be the hub for all of your automation tasks.

Features



Ansible Tower Features





Installation and configuration

1. Install package with - `yum install ansible` (or any other package manager for other os)
2. Add ansible user on all nodes with -

```
useradd ansible
```

```
passwd ansible
```

1. Add `ansible ALL=(ALL) NOPASSWD: ALL` to your `/etc/sudoers` file on control node.
2. Create ssh keys on control node and copy them to managed nodes with -

```
ssh-keygen
```

```
ssh-copy-id com.example1.cn
```




Grouping Nodes to inventory

Ansible needs an inventory file to perform operations on managed nodes.

- Create an inventory file - `vim /home/admin/inventory`
- Add managed nodes under a group. The nodes which we copied ssh keys in installation steps.

```
[webservers]
192.168.35.140
192.168.35.150

[datacache]
192.168.45.45

[appservers]
192.168.100.1
192.168.100.2

[dbservers]
172.35.0.10
```



Conclusion

You can group the servers in your inventory together by using group names. This grouping allows you to execute playbooks on only your web servers or, only on your database servers. You can also run ad-hoc commands. Ad-hoc commands are tasks that you need to run only once. For instance, if you needed to reboot all of your web servers. All in all, Ansible is an excellent tool you can use to save time, money and effort to automate tasks across a single or multiple remote servers.

But as new IAC tools are emerging ansible is lacking behind in terms the modern features and cloud infrastructure capabilities.

IAC tools: terraform, pulumi, cloudformation, ARM.