# Getting Started with Ansible

We will be using the following demo setup for learning Ansible. You can use Oracle Virtual Box, Vm Ware or can set up things on your cloud or Physical hardware.

**Note -** The document assumes Ubuntu 16.04 OS in action. For other OS and version, certain OS specific changed would be required.

# Setting up our Ansible demo Architecture

**Ansible Master** - Ubuntu 16.04/ 2 GB RAM /

**Host 1** - Ubuntu 16.04/ 1 GB RAM /

**Host 2** - Ubuntu 16.04/ 1 GB RAM /

## Installing Ansible on Ubuntu 16.04

Use the commands below to install Ansible

sudo apt-get update

sudo apt-get install ansible

You can also install ansible using pip. Follow the instructions for the same -

sudo apt-get update

sudo apt-get install python-pip pip install ansible

Fire command below to check installation

ansible --version

## Setting up the connection

Ansible needs SSH connection to hosts and that too, a passwordless ssh so that manual interuptions of typing passwords are ruled out. Follow the steps below to setup the servers -

Install openssh-server on all there instances - Ansible Master, Host1 and Host 2

sudo apt-get install openssh-server

sudo service ssh start

Generate keys on all 3 instances. Use the command below -

ssh-keygen

Now all our instances are SSH enabled. Try doing an SSH from Ansible to Host1 using the command below -

ssh username@ip-address-of-host-1

ssh username@ip-address-of-host-2

In my case the command becomes -

ssh [root@192.168.0.10](mailto:root@192.168.0.10)

ssh [root@192.168.0.11](mailto:root@192.168.0.11)

Enabling Passwordless access from ansbile to Host 1 and Host 2 Use the following command to enable passwordless access from Ansbile to hosts

ssh-copy-id [root@192.168.0.10](mailto:root@192.168.0.10)

ssh-copy-id [root@192.168.0.11](mailto:root@192.168.0.11)

Replace the above command with your username and ip address.

Try SSH again and it the connection should not request for a password.

## Introduction to Ansible key terms

**Target Hosts** are the machines which are to be controlled by ansible master

**Inventory File** is a plain file which contains the information of hosts which are to be maintained by Ansible master. This information is nothing but the domain

name, IP address of the target hosts.

**Ansible modules** Modules are special packages which are used to perform operations using ansible. For example, if you want to install vim in ubuntu, you use **apt** module of ansible. The complete repository of Ansible modules is [available here [https://docs.ansible.com/ansible/latest/modules/modules\_by\_category.html]](https://docs.ansible.com/ansible/latest/modules/modules_by_category.html)

**Ansible adhoc commands** are used for firing commands at run time. The are simiiar to that of shell commands with ansible driven syntax

**Ansible playbooks** are the scripted form of Ansible. It contains an ansbile script which can be executed multiple times. Just like a programming file.

Above given terms are enough to start with. Rest of them can be understood on the fly with examples.

## Setting up the inventory file

Inventory file has to contain the information of target hosts. The default inventory file for ansible is **/etc/ansible/hosts/** If you observe the contents of file, the example to write an inventory is given which can have hosts, ip adress and groups mentioned. Open /etc/ansible/hosts and the file is self-explainatory on how to add hosts.

Lets add our two hosts in inentory file -

vim /etc/ansible/hosts

192.168.0.10

192.168.0.11

## Running our first ansible command

ansible -m ping all

In the above given command we have

all - Use all defined servers from the inventory file

-m ping - Use the "ping" module, which simply runs the ping command and returns the results

The output of above command is -

* + - 1. | SUCCESS => {

"changed": false, "ping": "pong"

}

* + - 1. | SUCCESS => {

"changed": false, "ping": "pong"

}

Lets run another adhoc command to install nginx package

ansible all -s -m apt -a 'pkg=nginx state=installed update\_cache=true

In the command given above

all - Run on all defined hosts from the inventory file

-s - Run using sudo

-m apt - Use the apt module

-a 'pkg=nginx state=installed update\_cache=true' - Provide the arguments for the apt module, including the package name, our desired end state and whether to update the package repository cache or not

the output of the above command will look similiar to the one given below -

10.4.15.191 | SUCCESS => {

"cache\_update\_time": 1479354394, "cache\_updated": true, "changed": true,

"stderr": "",

"stdout": "Reading package lists...\nBuilding dependency tree...\nReading state information...\nnginx is already the newest version.\n0 upgraded, 0 newly installed, 0 to remove and 310 not upgraded.\n5 not fully installed or removed.\nAfter this operation, 0 B of additional disk space will be used.\nSetting up libjpeg- turbo8:i386 (1.1.90+svn733-0ubuntu4.4) ...\nSetting up libxslt1.1:i386 (1.1.26-8ubuntu1.3) ...\nSetting up nginx-common (1.1.19-1ubuntu0.8)

...\nSetting up nginx-light (1.1.19-1ubuntu0.8) ...\nSetting up nginx (1.1.19-1ubuntu0.8) ...\nProcessing triggers for libc-bin

...\nldconfig deferred processing now taking place\n",

"stdout\_lines": [

"Reading package lists...", "Building dependency tree...", "Reading state information...",

"nginx is already the newest version.",

"0 upgraded, 0 newly installed, 0 to remove and 310 not upgraded.",

"5 not fully installed or removed.",

"After this operation, 0 B of additional disk space will be used.",

"Setting up libjpeg-turbo8:i386 (1.1.90+svn733-0ubuntu4.4) ...", "Setting up libxslt1.1:i386 (1.1.26-8ubuntu1.3) ...",

"Setting up nginx-common (1.1.19-1ubuntu0.8) ...", "Setting up nginx-light (1.1.19-1ubuntu0.8) ...", "Setting up nginx (1.1.19-1ubuntu0.8) ...", "Processing triggers for libc-bin ...",

"ldconfig deferred processing now taking place"

]

}

## Ansible adhoc commands

The commands mentioned in the above section are the ad-hoc commands of ansible.

These commands to be fired again cannot be used unless you save it somewhere.

Ad-hoc commands are used to setup and manager servers for very rare actions.

A simple example can be to test connectivity using ping module of ansible.

## Issues with Adhoc commands

Adhoc commands are quick commands which are generally used to check system status like network, running status, etc. To do more meaningful auomation, we need to deal with more mature and easy to understand scripts which can be written in Ansible. Ansible playbooks come handy in such situations.