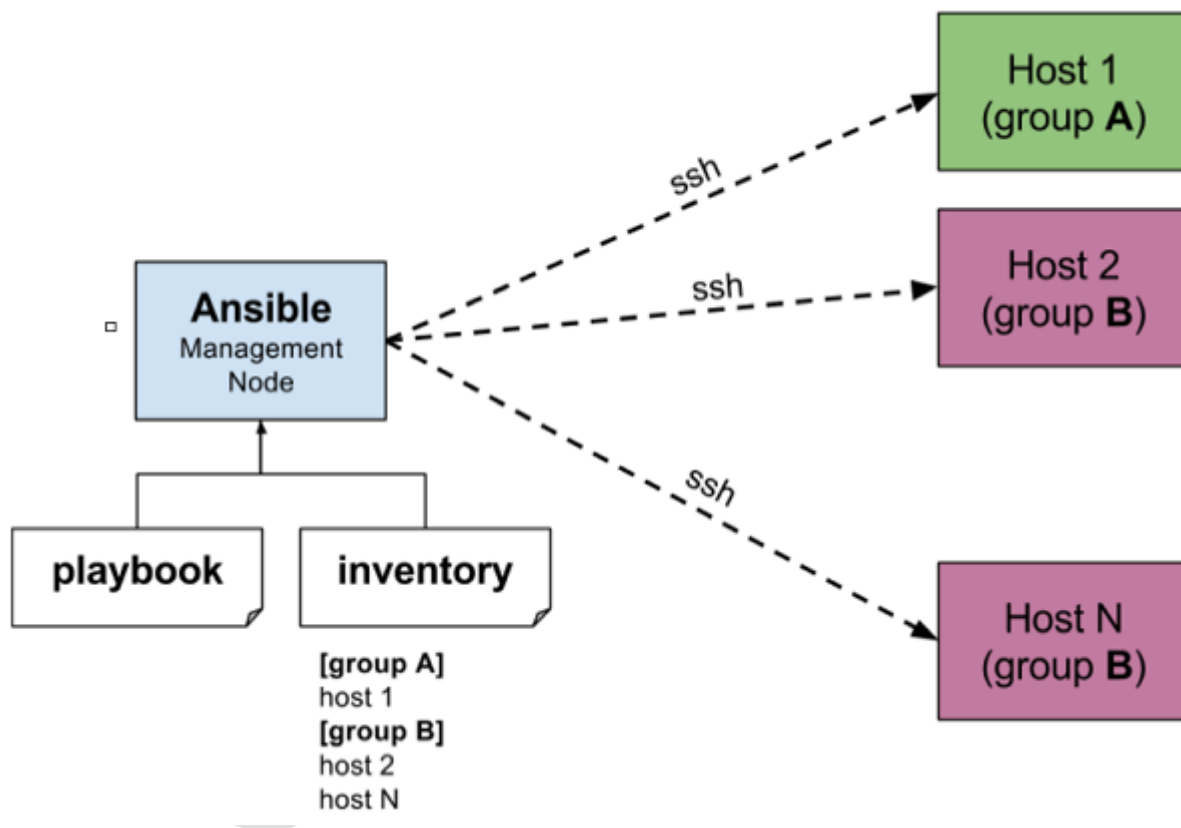


What is Ansible?

Ansible is an open source IT Configuration Management, Deployment & Orchestration tool. It aims to provide large productivity gains to a wide variety of automation challenges. This tool is very simple to use yet powerful enough to automate complex multi-tier IT application environments

Why we need Ansible?

We now use Ansible for any task or project that requires repeatable processes and a consistent environment, such as provisioning IoT devices and server infrastructure, installation and configuration of applications, and application deployment



When introduced Ansible?

- It is introduced in the year October 2015 the original author is Michael DeHaan
- But it is started in 1966
- It is written in python language

Chef – When there is a failure on the primary server i.e. chef server, it has a backup server to take the place of the primary server.



Puppet – It has multi-master architecture, if the active master goes down, the other master takes the active master place.

Ansible – It runs with a single active node, called the Primary instance. If primary goes down, there is a Secondary instance to take its place.

Salt stack – It can have multiple masters configured. If one master is down, agents connect with the other master in the list. Therefore it has multiple masters to configure salt minions.

Ease of Setup

Chef – Chef has a master-agent architecture.

Puppet – Puppet also has a master-agent architecture.

Ansible – It has only master running on the server machine, but no agents running on the client machine.

Salt stack – Here Server is called as salt master and clients are called as salt minions which run as agents in the client machine.

Configuration Language

Chef – Chef uses Ruby Domain Specific Language (Ruby DSL). It has a steep Learning Curve and its developer oriented.

Puppet

Puppet uses its own puppet Domain Specific Language (Puppet DSL). It is not very easy to learn and its system administrator oriented.

Ansible

Ansible uses YAML i.e yet another mark-up Language (Python). It is quite easy to learn and its administrator oriented. Python is inbuilt into most Unix and Linux deployments nowadays, so setting the tool up and running is quicker.

Salt stack



Salt stack also uses YAML (Python). It is again easy to learn and administrator oriented.

Machine should be

Chef – Chef Server works only on Linux/Unix but Chef Client and Workstation can be on windows as well.

Puppet – Puppet Master works only on Linux/Unix but Puppet Agent also works on windows.

Ansible – Ansible supports windows machines as well but the Ansible server has to be on Linux/Unix machine.

Salt stack – Salt Master works only on Linux/Unix but Salt minions can work on windows as well.

Chef - push

Puppet-pull

Ansible-push

Salt stack-push & pull

Ansible topics

- Ad hoc command
- Play books
- Roles
- Vault

Ad hoc command is used to install a single task

Play books is used to install and doing multiple task using YAML lag

Roles is dividing a complex playbooks into a structural format

Vault is used to hide the your secret files



In this we going to see ansible

Step 1: ANSIBLE INSTALLATION

Launch the EC2 instance

The screenshot shows the AWS Management Console interface. At the top, there's a navigation bar with 'Resource Groups', a search bar, and user information 'nishanthanravi1993'. Below the navigation bar, there's a 'Launch Instance' button and a 'Connect' button. A table lists EC2 instances with columns: Name, Instance ID, Instance Type, Availability Zone, Instance State, Status Checks, Alarm Status, and Public DNS (IPv4). The table shows three instances: 'Ansible master' (terminated), 'ansible' (running), and another 'ansible' (running). The details for the 'ansible' instance (ID: i-0dc50483e8c34394f) are shown below the table. The details include Instance ID, Instance state (running), Instance type (t2.micro), Elastic IPs, Availability zone (ap-south-1a), Security groups (launch-wizard-14), Public DNS (IPv4) (ec2-13-127-37-191.ap-south-1.compute.amazonaws.com), IPv4 Public IP (13.127.37.191), IPv6 IPs (-), Private DNS (ip-172-31-44-215.ap-south-1.compute.internal), Private IPs (172.31.44.215), and Secondary private IPs.

Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks	Alarm Status	Public DNS (IPv4)
Ansible master	i-0147eb4b9e4b346...	t2.micro	ap-south-1b	terminated		None	
ansible	i-038a9c697c3dd3fe2	t2.micro	ap-south-1a	running	Initializing	None	ec2-13-233-127...
ansible	i-05356596a42a62a...	t2.micro	ap-south-1a	running	Initializing	None	ec2-35-154-22...
ansible	i-0dc50483e8c34394f	t2.micro	ap-south-1a	running	Initializing	None	ec2-13-127-37...

Instance: **i-0dc50483e8c34394f (ansible)** Public DNS: **ec2-13-127-37-191.ap-south-1.compute.amazonaws.com**

Description	
Instance ID	i-0dc50483e8c34394f
Instance state	running
Instance type	t2.micro
Elastic IPs	
Availability zone	ap-south-1a
Security groups	launch-wizard-14. view inbound rules . view outbound rules
Public DNS (IPv4)	ec2-13-127-37-191.ap-south-1.compute.amazonaws.com
IPv4 Public IP	13.127.37.191
IPv6 IPs	-
Private DNS	ip-172-31-44-215.ap-south-1.compute.internal
Private IPs	172.31.44.215
Secondary private IPs	

Step 2: User data:

```
#!/bin/bash
```

```
yum install python-pip -y
```

```
pip install ansible
```

e Details

- ☐ Protect against accidental termination
- ☐ Enable CloudWatch detailed monitoring
Additional charges apply.
- Additional charges will apply for dedicated tenancy.

- ☐ Enable
Additional charges may apply

- ☒ As text ☐ As file ☐ Input is already base64 encoded

```
#!/bin/bash
yum install python-pip -y
pip install ansible
```

Cancel Previous **Review and Launch** Next: Add Storage

Step 3: Download Pageagent and load your ppk file.

Using this Url

<https://www.chiark.greenend.org.uk/~sgtatham/putty/latest.html>

And goto this page

← → ↻ [chiark.greenend.org.uk/~sgtatham/putty/latest.html](https://www.chiark.greenend.org.uk/~sgtatham/putty/latest.html) ☆ ⓘ ⋮

Apps Google YouTube Maps

Download PuTTY: latest release (0.73)

[Home](#) | [FAQ](#) | [Feedback](#) | [Licence](#) | [Updates](#) | [Mirrors](#) | [Keys](#) | [Links](#) | [Team](#)
Download: [Stable](#) · [Snapshot](#) | [Docs](#) | [Changes](#) | [Wishlist](#)

This page contains download links for the latest released version of PuTTY. Currently this is 0.73, released on 2019-09-29.

When new releases come out, this page will update to contain the latest, so this is a good page to bookmark or link to. Alternatively, here is a [permanent link to the 0.73 release](#).

Release versions of PuTTY are versions we think are reasonably likely to work well. However, they are often not the most up-to-date version of the code available. If you have a problem with this release, then it might be worth trying out the [development snapshots](#), to see if the problem has already been fixed in those versions.

Package files

You probably want one of these. They include versions of all the PuTTY utilities.
(Not sure whether you want the 32-bit or the 64-bit version? Read the [FAQ entry](#).)

MSI ("Windows Installer")

32-bit:	putty-0.73-installer.msi	(or by FTP)	(signature)
64-bit:	putty-64bit-0.73-installer.msi	(or by FTP)	(signature)

Unix source archive

.tar.gz:	putty-0.73.tar.gz	(or by FTP)	(signature)
----------	-----------------------------------	-------------	-------------

Alternative binary files

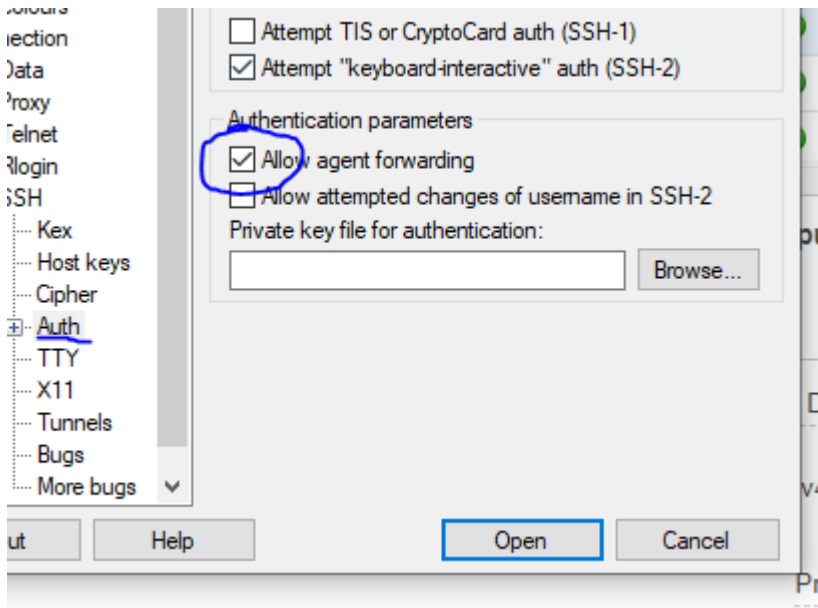
IPv6 IPs -

Private DNS ip-172-31-44-215.ap-south-1.compute.internal

Private IPs 172.31.44.215

Secondary private IPs

Using right click add the ppk file in the Pageagent



Step 4: In putty ssh session enable allow agent forwarding option- Otherwise while connecting to node instance you will get permission denied error

Logon to ec2-user

```
login as: ec2-user
Authenticating with public key "imported-openssh-key" from agent

  ____|_  ____|_  )
  ____|_  ____|_  /   Amazon Linux 2 AMI
  ____|_  ____|_  |

https://aws.amazon.com/amazon-linux-2/
15 package(s) needed for security, out of 31 available
Run "sudo yum update" to apply all updates.
[ec2-user@ip-172-31-38-107 ~]$
```

Step 5: Ansible adhoc command: Practice the command below with ec2-user and not with root user

Step 6: Create one text file for e.g. slaves.txt and add node instance private IP

```
15 package(s) needed for security, out of 31 available
Run "sudo yum update" to apply all updates.
[ec2-user@ip-172-31-38-107 ~]$ vi slaves.txt
```

Example:

[Web]

IP 1

IP 2



Step 7: Now you need ansible.cfg file and u can get it from

Using this Url

<https://raw.githubusercontent.com/ansible/ansible/devel/examples/ansible.cfg>

```
[ec2-user@ip-172-31-38-107 ~]$ wget https://raw.githubusercontent.com/ansible/ansible/devel/examples/ansible.cfg
--2019-10-16 04:28:46-- https://raw.githubusercontent.com/ansible/ansible/devel/examples/ansible.cfg
Resolving raw.githubusercontent.com (raw.githubusercontent.com)... 151.101.152.133
Connecting to raw.githubusercontent.com (raw.githubusercontent.com)|151.101.152.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 19985 (20K) [text/plain]
Saving to: 'ansible.cfg'

100%[=====>] 19,985 --.-K/s in 0s

2019-10-16 04:28:46 (104 MB/s) - 'ansible.cfg' saved [19985/19985]

[ec2-user@ip-172-31-38-107 ~]$
```

Vi ansible.cfg

```
# ohai - import facts from ohai
# You can combine them using comma (ex: network,virtual)
# You can negate them using ! (ex: !hardware,!factor,!ohai)
# A minimal set of facts is always gathered.
#gather_subset = all

# some hardware related facts are collected
# with a maximum timeout of 10 seconds. This
# option lets you increase or decrease that
# timeout to something more suitable for the
# environment.
# gather_timeout = 10

# Ansible facts are available inside the ansible_facts.* dictionary
# namespace. This setting maintains the behaviour which was the default prior
# to 2.5, duplicating these variables into the main namespace, each with a
# prefix of 'ansible_'.
# This variable is set to True by default for backwards compatibility. It
# will be changed to a default of 'False' in a future release.
# ansible_facts.
# inject_facts_as_vars = True

# additional paths to search for roles in, colon separated
#roles_path = /etc/ansible/roles

# uncomment this to disable SSH key host checking
host_key_checking = False

# change the default callback, you can only have one 'stdout' type enabled at a time.
#stdout_callback = skippy

-- INSERT --
```

71,1

9%

Step 8:

ansible all -i slaves.txt -m ping

```
[web]
172.31.41.255
172.31.44.215
~
```

ansible web -i slaves.txt -m ping

```
[ec2-user@ip-172-31-38-107 ~]$ vi ansible.cfg
[ec2-user@ip-172-31-38-107 ~]$ ansible web -i slaves.txt -m ping
[WARNING]: Platform linux on host 172.31.44.215 is using the discovered Python interpreter at /usr/bin/python, but
future installation of another Python interpreter could change this. See
https://docs.ansible.com/ansible/2.8/reference_appendices/interpreter_discovery.html for more information.

172.31.44.215 | SUCCESS => {
  "ansible_facts": {
    "discovered_interpreter_python": "/usr/bin/python"
  },
  "changed": false,
  "ping": "pong"
}
[WARNING]: Platform linux on host 172.31.41.255 is using the discovered Python interpreter at /usr/bin/python, but
future installation of another Python interpreter could change this. See
https://docs.ansible.com/ansible/2.8/reference_appendices/interpreter_discovery.html for more information.

172.31.41.255 | SUCCESS => {
  "ansible_facts": {
    "discovered_interpreter_python": "/usr/bin/python"
  },
  "changed": false,
  "ping": "pong"
}
[ec2-user@ip-172-31-38-107 ~]$
```

Step 9:

Ansible web -i slaves.txt -m yum -a "name=httpd state=present" -b

```
[ec2-user@ip-172-31-38-107 ~]$ ansible web -i slaves.txt -m yum -a "name=httpd state=present" -b
[WARNING]: Platform linux on host 172.31.41.255 is using the discovered Python interpreter at /usr/bin/python, but
future installation of another Python interpreter could change this. See
https://docs.ansible.com/ansible/2.8/reference_appendices/interpreter_discovery.html for more information.

172.31.41.255 | CHANGED => {
  "ansible_facts": {
    "discovered_interpreter_python": "/usr/bin/python"
  },
  "changed": true,
  "changes": {
    "installed": [
      "httpd"
    ]
  },
  "msg": "",
  "rc": 0,
  "results": {

```

Step 10:

Ansible web -i slaves.txt -m service -a "name=httpd state=started" -b

```
[ec2-user@ip-172-31-38-107 ~]$ ansible web -i slaves.txt -m service -a "name=httpd state=started" -b
[WARNING]: Platform linux on host 172.31.41.255 is using the discovered Python interpreter at /usr/bin/python, but future
installation of another Python interpreter could change this. See
https://docs.ansible.com/ansible/2.8/reference_appendices/interpreter_discovery.html for more information.

172.31.41.255 | CHANGED => {
  "ansible_facts": {
    "discovered_interpreter_python": "/usr/bin/python"
  },
  "changed": true,
  "name": "httpd",
  "state": "started",
  "status": {
    "ActiveEnterTimestampMonotonic": "0",
    "ActiveExitTimestampMonotonic": "0",
    "ActiveState": "inactive",
    "After": "remote-fs.target system.slice httpd-init.service nss-lookup.target -.mount tmp.mount network.target basic.target sys
temd-journald.socket",
    "AllowIsolate": "no",
    "AmbientCapabilities": "0",
    "AssertResult": "no",
    "AssertTimestampMonotonic": "0",
    "Before": "shutdown.target",

```


Step 11:

Transferring the file to many servers/machines

ansible all -i slaves.txt -m copy -a "src=./slaves.txt dest=/tmp/slaves.txt"

```
[ec2-user@ip-172-31-38-107 ~]$ ansible all -i slaves.txt -m copy -a "src=./slaves.txt dest=/tmp/slaves.txt"
[WARNING]: Platform linux on host 172.31.41.255 is using the discovered Python interpreter at /usr/bin/python, but future installation of another
Python interpreter could change this. See https://docs.ansible.com/ansible/2.8/reference_appendices/interpreter_discovery.html for more information.
172.31.41.255 | CHANGED => {
  "ansible_facts": {
    "discovered_interpreter_python": "/usr/bin/python"
  },
  "changed": true,
  "checksum": "591224e514ed3b4b58a2250b9c332c126214a691",
  "dest": "/tmp/slaves.txt",
  "gid": 1000,
  "group": "ec2-user",
  "md5sum": "2ebcee162856d8f1898d63309ed9c664",
  "mode": "0664",
  "owner": "ec2-user",
  "size": 34,
  "src": "/home/ec2-user/.ansible/tmp/ansible-tmp-1571214270.93-72194836138402/source",
  "state": "file",
  "uid": 1000
}
```

Setp 12:

Deleting whole directory and files

Ansible abc -m file -a "dest = /path/user1/new state = absent"

Step 13: (for Ubuntu machine)

For installation and management of applications

ansible webserver -m apt -a 'name=python state=present'

Step 14:

User: To add and delete users

ansible webserver -m user -a 'name=nishanth password=admin123' -b

ansible webserver -m user -a 'name=nishanth state=absent' (to delete user) -b

```

[ec2-user@ip-172-31-38-107 ~]$ ansible all -i slaves.txt -m user -a "name=nishanth password=admin123" -b
[WARNING]: The input password appears not to have been hashed. The 'password' argument must be encrypted for this module to work properly.

[WARNING]: Platform linux on host 172.31.41.255 is using the discovered Python interpreter at /usr/bin/python, but future installation of another Python interpreter could change this. See https://docs.ansible.com/ansible/2.8/reference_appendices/interpreter_discovery.html for more information.

172.31.41.255 | CHANGED => {
  "ansible_facts": {
    "discovered_interpreter_python": "/usr/bin/python"
  },
  "changed": true,
  "comment": "",
  "create_home": true,
  "group": 1001,
  "home": "/home/nishanth",
  "name": "nishanth",
  "password": "NOT_LOGGING_PASSWORD",
  "shell": "/bin/bash",
  "state": "present",
  "system": false,
  "uid": 1001
}

```

Task: use Ubuntu machine and without using -i

Step 15:

Deploy your webapp straight from git:

```
ansible webservers -m git -a "repo=https://foo.example.org/repo.git
dest=/srv/myapp version=HEAD"
```

```
ansible webservers -m service -a "name=httpd state=restarted"
```

```
ansible all -i slaves.txt --list-host
```

```
ansible <specific ip> -i slaves.txt -a "uname -a"
```

```
ansible all -i slaves.txt -a "uname -a" -u ec2-user
```

```
ansible all -i slaves.txt -a "uname -a" -u ec2-user -b
```

```
ansible jenkins -i slaves.txt -a "grep -i JENKINS_PORT
/etc/sysconfig/jenkins" -b
```

Dynamic Inventory:

Step 1:

You can find sample python script for dynamic inventory in this URL

<https://raw.githubusercontent.com/ansible/ansible/devel/contrib/inventory/ec2.py>

← → ↻ raw.githubusercontent.com/ansible/ansible/dev/contrib/inventory/ec2.py ☆ ⓘ ⋮

Apps Google YouTube Maps

```
#!/usr/bin/env python
...
EC2 external inventory script
=====

Generates inventory that Ansible can understand by making API request to
AWS EC2 using the Boto library.

NOTE: This script assumes Ansible is being executed where the environment
variables needed for Boto have already been set:
    export AWS_ACCESS_KEY_ID='AKI123'
    export AWS_SECRET_ACCESS_KEY='abc123'

Optional region environment variable if region is 'auto'

This script also assumes that there is an ec2.ini file alongside it. To specify a
different path to ec2.ini, define the EC2_INI_PATH environment variable:

    export EC2_INI_PATH=/path/to/my_ec2.ini

If you're using euca2ools you need to set the above variables and
you need to define:

    export EC2_URL=http://hostname_of_your_cc:port/services/Eucalyptus

If you're using boto profiles (requires boto>=2.24.0) you can choose a profile
using the --boto-profile command line argument (e.g. ec2.py --boto-profile prod) or using
the AWS_PROFILE variable:

    AWS_PROFILE=prod ansible-playbook -i ec2.py myplaybook.yml

For more details, see: http://docs.pythonboto.org/en/latest/boto_config_tut.html

You can filter for specific EC2 instances by creating an environment variable
named EC2_INSTANCE_FILTERS, which has the same format as the instance_filters
entry documented in ec2.ini. For example, to find all hosts whose name begins
with 'webserver', one might use:

    export EC2_INSTANCE_FILTERS='tag:Name=webserver'
```

```
[ec2-user@ip-172-31-38-107 ~]$ wget https://raw.githubusercontent.com/ansible/ansible/dev/contrib/inventory/ec2.py
--2019-10-16 08:30:24-- https://raw.githubusercontent.com/ansible/ansible/dev/contrib/inventory/ec2.py
Resolving raw.githubusercontent.com (raw.githubusercontent.com)... 151.101.152.133
Connecting to raw.githubusercontent.com (raw.githubusercontent.com)|151.101.152.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 73130 (71K) [text/plain]
Saving to: 'ec2.py'

100%[=====>] 73,130 --.-K/s in 0.002s

2019-10-16 08:30:24 (32.9 MB/s) - 'ec2.py' saved [73130/73130]

[ec2-user@ip-172-31-38-107 ~]$
```

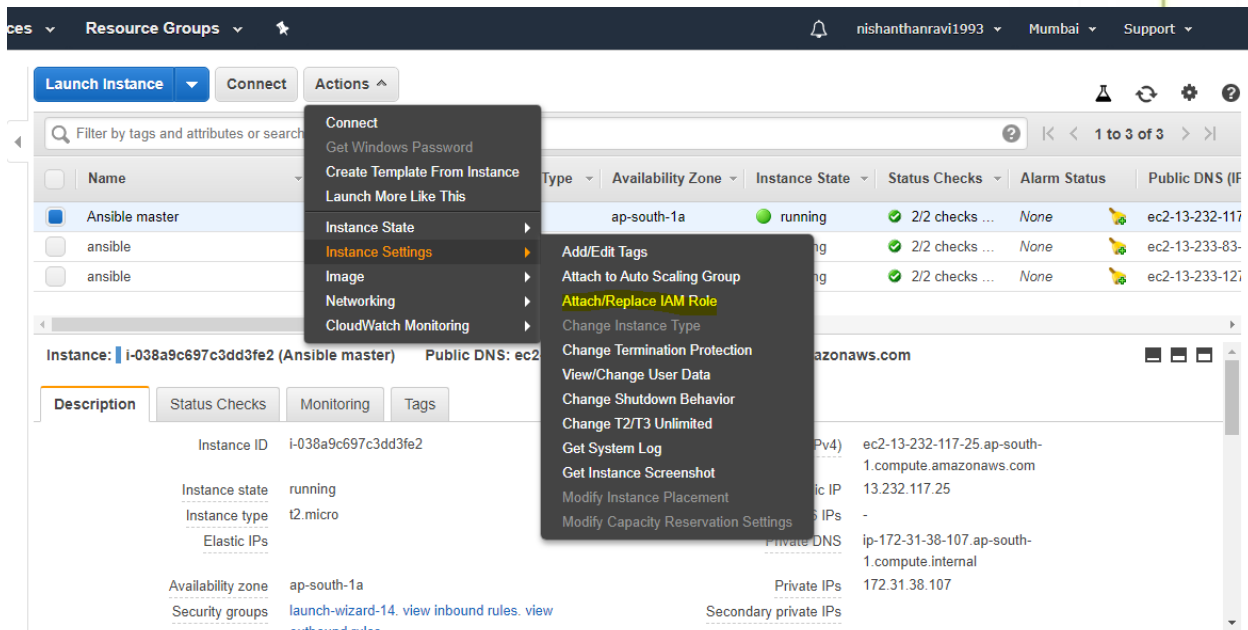
chmod 777 ec2.py

```
2019-10-16 08:30:24 (32.9 MB/s) - 'ec2.py' saved [73130/73130]

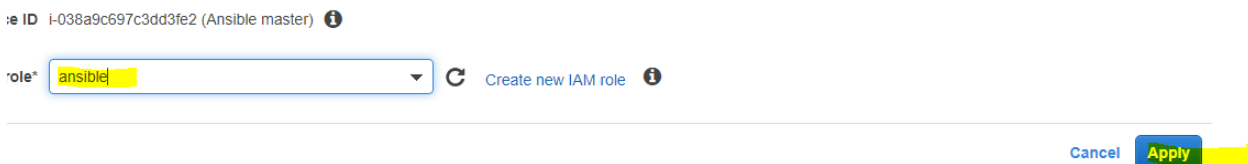
[ec2-user@ip-172-31-38-107 ~]$ chmod 777 ec2.py
[ec2-user@ip-172-31-38-107 ~]$
```

Check ls -lrt

Step 2:



Using IAM full admin access



Install boto in control machine to work with dynamic inventory with the below command.

Pip install boto

```
[ec2-user@ip-172-31-38-107 ~]$ ansible all -i ec2.py -m ping
[DEPRECATION WARNING]: The TRANSFORM_INVALID_GROUP_CHARS settings is set to allow bad characters in group names by default, this will change, but
still be user configurable on deprecation. This feature will be removed in version 2.10. Deprecation warnings can be disabled by setting
deprecation_warnings=False in ansible.cfg.
[WARNING]: Invalid characters were found in group names but not replaced, use -vvvv to see details

[WARNING]: Platform linux on host 13.233.83.10 is using the discovered Python interpreter at /usr/bin/python, but future installation of another
Python interpreter could change this. See https://docs.ansible.com/ansible/2.8/reference_appendices/interpreter_discovery.html for more information.
13.233.83.10 | SUCCESS => {
  "ansible_facts": {
    "discovered_interpreter_python": "/usr/bin/python"
  },
  "changed": false,
  "ping": "pong"
}

[WARNING]: Platform linux on host 13.233.127.134 is using the discovered Python interpreter at /usr/bin/python, but future installation of another
Python interpreter could change this. See https://docs.ansible.com/ansible/2.8/reference_appendices/interpreter_discovery.html for more information.
13.233.127.134 | SUCCESS => {
  "ansible_facts": {
    "discovered_interpreter_python": "/usr/bin/python"
  },
  "changed": false,
  "ping": "pong"
}

[WARNING]: Platform linux on host 13.232.117.25 is using the discovered Python interpreter at /usr/bin/python, but future installation of another
Python interpreter could change this. See https://docs.ansible.com/ansible/2.8/reference_appendices/interpreter_discovery.html for more information.
13.232.117.25 | SUCCESS => {
  "ansible_facts": {
    "discovered_interpreter_python": "/usr/bin/python"
  },
  "changed": false,
  "ping": "pong"
}
[ec2-user@ip-172-31-38-107 ~]$
```

Step 3:

Create IAM role in AWS with full administrative access and attach role to your control machine

Step 4:

Dynamic Inventory example

```
ansible all -i ec2.py -a "uname -a"
```



Playbooks

Step 1:

Install visual studio code

<https://code.visualstudio.com/docs/?dv=win64user>

Step 2:

Install ansible plugins in visual studio code which will make our job easy

Step 3:

playbook file should be with .yaml format

vi filename.yaml

```
- hosts: all
  remote_user: ec2-user
  become: yes
  tasks:
    - name: im gonig to install apache
      yum:
        name: httpd
        state: present
    - name: start the service
      service:
        name: httpd
        state: started
    - name: configure index page
      copy:
        src: index.html
        dest: /var/www/html/index.html
    - name: just a test
```

To run

```
ansible-playbook -l slaves.txt filename.yaml
```

```
[ec2-user@ip-172-31-38-107 ~]$ vi fourth.yaml
[ec2-user@ip-172-31-38-107 ~]$ ansible-playbook -i slaves.txt fourth.yaml

PLAY [all] *****

TASK [Gathering Facts] *****
ok: [172.31.41.255]
ok: [172.31.44.215]

TASK [im gonig to install apache] *****
ok: [172.31.41.255]
ok: [172.31.44.215]

TASK [start the service] *****
ok: [172.31.41.255]
ok: [172.31.44.215]

TASK [configure index page] *****
changed: [172.31.41.255]
changed: [172.31.44.215]

PLAY RECAP *****
172.31.41.255      : ok=4    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
172.31.44.215      : ok=4    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0

[ec2-user@ip-172-31-38-107 ~]$
```

Task 1:

- hosts: all

remote_user: ec2-user

become: yes

tasks:

- name: install httpd server

package:

name: httpd

state: present

- name: Start service httpd, if not running

service:

name: httpd

state: started

- template:

src: /home/ec2-user/index.html.j2

dest: /var/www/html/index.html

To run this file

Task 2:



Create the (vi file name.yaml)

```
- hosts: all
  remote_user: ec2-user
  become: yes
  tasks:
    - name: im going to install apache
      yum:
        name: "{{ item }}"
        state: present
      with_items:
        - mysql
        - php
        - unzip
```

Run this (ansible-playbook -i slaves.txt filename.yaml)

Task 3:



```
PLAY [web] *****

TASK [Gathering Facts] *****
ok: [172.31.41.255]
ok: [172.31.44.215]

TASK [Add jenkins repo install] *****
ok: [172.31.41.255]
ok: [172.31.44.215]

TASK [jenkins java install] *****
fatal: [172.31.41.255]: FAILED! => ("msg": "The task includes an option with an undefined variable. The error was: 'items' is undefined\n\nThe error appears to be in '/home/ec2-user/seventh.yaml': line 11, column 7, but maybe elsewhere in the file depending on the exact syntax problem.\n\nThe offending line appears to be:\n\n  gpgkey: https://pkg.jenkins.io/redhat-stable/jenkins.io.key\n    ~ name: jenkins java install\n    ~ here\n"),
fatal: [172.31.44.215]: FAILED! => ("msg": "The task includes an option with an undefined variable. The error was: 'items' is undefined\n\nThe error appears to be in '/home/ec2-user/seventh.yaml': line 11, column 7, but maybe elsewhere in the file depending on the exact syntax problem.\n\nThe offending line appears to be:\n\n  gpgkey: https://pkg.jenkins.io/redhat-stable/jenkins.io.key\n    ~ name: jenkins java install\n    ~ here\n")

PLAY RECAP *****
172.31.41.255      : ok=2    changed=0    unreachable=0    failed=1    skipped=0    rescued=0    ignored=0
172.31.44.215      : ok=2    changed=0    unreachable=0    failed=1    skipped=0    rescued=0    ignored=0

[ec2-user@ip-172-31-38-107 ~]$ vi seventh.yaml
[ec2-user@ip-172-31-38-107 ~]$ ansible-playbook -i slaves.txt seventh.yaml

PLAY [web] *****

TASK [Gathering Facts] *****
ok: [172.31.41.255]
ok: [172.31.44.215]

TASK [Add jenkins repo install] *****
ok: [172.31.41.255]
ok: [172.31.44.215]

TASK [jenkins java install] *****
changed: [172.31.41.255] => (item=java)
changed: [172.31.44.215] => (item=java)
```

- hosts: web

remote_user: ec2-user

become: yes

tasks:

- name: Add jenkins repo install

yum_repository:

name: jenkins

description: jenkins YUM repo

baseurl: https://pkg.jenkins.io/redhat-stable

gpgkey: https://pkg.jenkins.io/redhat-stable/jenkins.io.key

- name: jenkins java install

yum:

name: "{{ item }}"

state: present

loop:

- java

- jenkins

- name: jenkins start

service:

name: jenkins

state: started



```
TASK [Add jenkins repo install] *****
ok: [172.31.41.255]
ok: [172.31.44.215]

TASK [jenkins java install] *****
changed: [172.31.41.255] => (item=java)
changed: [172.31.44.215] => (item=java)
changed: [172.31.41.255] => (item=jenkins)
changed: [172.31.44.215] => (item=jenkins)

TASK [jenkins start] *****
changed: [172.31.41.255]
changed: [172.31.44.215]

PLAY RECAP *****
172.31.41.255      : ok=4    changed=2    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
172.31.44.215      : ok=4    changed=2    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0

[ec2-user@ip-172-31-38-107 ~]$
```

Change the port num for jenkins

```
- hosts: web
  remote_user: ec2-user
  become: yes
  vars:
    port: 8000
  tasks:
    - name: Add jenkins repo install
      yum_repository:
        name: jenkins
        description: jenkins YUM repo
        baseurl: https://pkg.jenkins.io/redhat-stable
        gpgkey: https://pkg.jenkins.io/redhat-stable/jenkins.io.key
    - name: jenkins java install
      yum:
        name: "{{ item }}"
        state: present
        loop:
          - java
          - jenkins
    - name: going to change port
      lineinfile:
        path: /etc/sysconfig/jenkins
        regexp: '^jenkins-port='
        line: "jenkins-port={{port}}"
        notify:
          - restart jenkins
    - name: jenkins start
      service:
        name: jenkins
        state: started
  handlers:
    - name: restart jenkins
      service:
        name: jenkins
        state: restarted
```

- hosts: web

remote_user: ec2-user

become: yes

vars:

port: 8000

tasks:

- name: Add jenkins repo install

- yum_repository:

- name: jenkins

- description: jenkins YUM repo

- baseurl: <https://pkg.jenkins.io/redhat-stable>

- gpgkey: <https://pkg.jenkins.io/redhat-stable/jenkins.io.key>

- name: jenkins java install

- yum:

- name: "{{ item }}"

- state: present

- loop:

- java

- jenkins

- name: going to change port

- lineinfile:

- path: /etc/sysconfig/jenkins

- regexp: '^jenkins-port='

- line: "jenkins-port-{{port}}"

- notify:

- restart jenkins

- name: jenkins start

- service:

- name: jenkins

- state: started

- handlers:

- name: restart jenkins

service:

name: jenkins

state: restarted



Task 4:

- hosts: jenkins

remote_user: ec2-user

become: yes

vars:

jenkins_port: 9006

tasks:

- name: jenkins installation from yum

yum_repository:

name: jenkins

description: jenkins

baseurl: <http://pkg.jenkins.io/redhat>

gpgkey: <https://jenkins-ci.org/redhat/jenkins-ci.org.key>

- name : Install jenkins and Java

package:

name: "{{item}}"

state: present

loop:

- java

- jenkins

- httpd

- name: we are going to start the service

service:

name: jenkins # required. Name of the service.

enabled: yes # not required. Whether the service should start on boot.

B(At least one of state and enabled are required.)

state: started

- name: change default port number

lineinfile:

path: /etc/sysconfig/jenkins

regexp: '^JENKINS_PORT='

line: "JENKINS_PORT={{ jenkins_port }}"

notify:

- restart jenkins

- name: validate port change

command: "grep -i JENKINS_PORT /etc/sysconfig/jenkins"

register: grep_results

- name: debug grep results

debug:

msg: "{{ hostvars['172.31.21.72'] }}"

when: grep_results.rc != 0

handlers:

- name: restart jenkins

service:

name: jenkins # required. Name of the service.

state: restarted

Task 5: Run playbook:

`ansible-playbook -i slaves.txt jenkins.yml`

Example to pass variable in run time

`ansible-playbook -i slaves.txt jenkins.yml -e "jenkins_port=8001"`

Roles:

Step 1:

Sudo yum install tree -y

`ec2-user@ip-172-31-2-249:~`

```
[ec2-user@ip-172-31-2-249 ~]$ sudo yum install tree -y
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
Package tree-1.6.0-10.amzn2.0.1.x86_64 already installed and latest version
Nothing to do
[ec2-user@ip-172-31-2-249 ~]$
```

Now create new file

```
[ec2-user@ip-172-31-2-249 ~]$ ansible-galaxy init rolestest1
- Role rolestest1 was created successfully
[ec2-user@ip-172-31-2-249 ~]$
```

Now log into file

```
[ec2-user@ip-172-31-2-249 ~]$ cd rolestest1
[ec2-user@ip-172-31-2-249 rolestest1]$ ls -lrt
total 4
drwxrwxr-x 2 ec2-user ec2-user  6 Nov  5 11:03 templates
-rw-rw-r-- 1 ec2-user ec2-user 1328 Nov  5 11:03 README.md
drwxrwxr-x 2 ec2-user ec2-user  6 Nov  5 11:03 files
drwxrwxr-x 2 ec2-user ec2-user 22 Nov  5 11:03 defaults
drwxrwxr-x 2 ec2-user ec2-user 22 Nov  5 11:03 tasks
drwxrwxr-x 2 ec2-user ec2-user 22 Nov  5 11:03 meta
drwxrwxr-x 2 ec2-user ec2-user 22 Nov  5 11:03 handlers
drwxrwxr-x 2 ec2-user ec2-user 22 Nov  5 11:03 vars
drwxrwxr-x 2 ec2-user ec2-user 39 Nov  5 11:03 tests
[ec2-user@ip-172-31-2-249 rolestest1]$
```

Cd task/

Using vi main.yml

Check it current dictory

```
[ec2-user@ip-172-31-2-249 tasks]$ vi main.yml
[ec2-user@ip-172-31-2-249 tasks]$ pwd
/home/ec2-user/rolestest/tasks
[ec2-user@ip-172-31-2-249 tasks]$ cd ../files/
[ec2-user@ip-172-31-2-249 files]$
```

Using **cd**

come back path

vi role.yml

ec2-user@ip-172-31-2-249:~

```
- hosts: all
  remote_user: ec2-user
  become: yes
  roles:
    - rolestest
```

```
[ec2-user@ip-172-31-2-249 ~]$ vi roles.yaml
[ec2-user@ip-172-31-2-249 ~]$ ansible-playbook -i slaves.txt roles.yaml

PLAY [all] *****

TASK [Gathering Facts] *****
ok: [13.232.36.128]

TASK [rolestest : im going to install apache] *****
ok: [13.232.36.128]

TASK [rolestest : start service] *****
ok: [13.232.36.128]

TASK [rolestest : configure index page] *****
ok: [13.232.36.128]

PLAY RECAP *****
13.232.36.128      : ok=4    changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0

[ec2-user@ip-172-31-2-249 ~]$
```

```
[ec2-user@ip-172-31-2-249 ~]$ tree rolestest
```

```
rolestest
├── defaults
│   └── main.yml
├── files
│   ├── index.html
│   └── roles.yaml
├── handlers
│   └── main.yml
├── meta
│   └── main.yml
├── README.md
├── tasks
│   └── main.yml
├── templates
├── tests
│   ├── inventory
│   └── test.yml
└── vars
    └── main.yml
```

8 directories, 10 files

```
[ec2-user@ip-172-31-2-249 ~]$
```