How to launch Amazon EC2 instances and configure Apache HTTP servers using Ansible Playbook

What is Amazon EC2 instance ?

Amazon Elastic Compute Cloud (Amazon EC2) provides scalable computing capacity in the Amazon Web Services (AWS) cloud. Using Amazon EC2 eliminates your need to invest in hardware upfront, so you can develop and deploy applications faster. You can use Amazon EC2 to launch as many or as few virtual servers as you need, configure security and networking, and manage storage. Amazon EC2 enables you to scale up or down to handle changes in requirements or spikes in popularity, reducing your need to forecast traffic.

What is Ansible ?

Ansible is an open source software that automates software provisioning, configuration management, and application deployment. Ansible connects via SSH, remote PowerShell or via other remote APIs.

Now, our problem, launching ec2 instance and deploying web server automatically, using one of the most popular configure management tool, ***Ansible***, can be done in two methods,

1. Simple playbook without defining any roles (with just playbook yaml alone)
2. Structured playbook with appropriate roles (Create EC2 & Deploy Apache)

This document discuss both the methods in detail with each and every step. You may follow one of the method whichever method is feasible and comfortable to you. However, the output of both the methods is same post execution of your play, EC2 instances are created in your AWS account and are deployed with Apache web server!

Method 1

In this method, we define all the required tasks into a single yaml playbook file, using which we run our play with Ansible playbook command to create ec2 instance and deploy apache.

The below steps are to be performed on any Linux machine (Ansible hosted server) preferably Amazon Linux, CentOS, & RHEL distributions. If you are running on Ubuntu or Debian based, there might be few changes in installing packages (apt-get inplace of yum for example).

⇒ Step 1: *Install dependent packages*

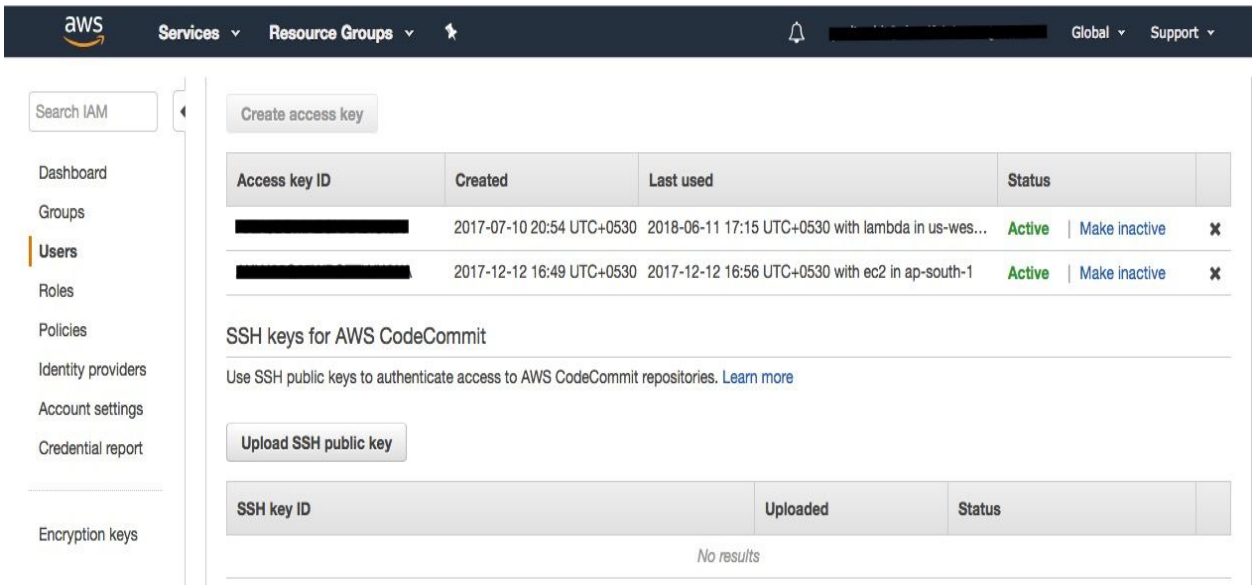
Run the following commands to install the required dependencies for Ansible and AWS.

```
sudo yum install python pip
sudo pip install --upgrade pip
sudo pip install boto
sudo yum install ansible
```

⇒ Step 2: *Gather AWS Configuration from AWS console*

Log into your EC2 Management Console (AWS) account to get your *AWS_ACCESS_KEY_ID* and *AWS_SECRET_ACCESS_KEY*. Go to “Identity and Access Management”. Create a new user if there isn’t one or select an existing user. Go to “Security Credentials” and click “Create Access Key”. Here’s an example of what you’ll end up with:

Access Key ID: XXXXXXXXXXXX
Secret Access Key: XXXXXXXXXXXXXXXXX



The basic requirements that need to be initialized to launch an EC2 instance are,

- aws access key => AWS access key id of the IAM user (see above)
- aws secret key => AWS access secret key of the IAM user (see above)
- region => The region in which the instance needs to be launched.
- security group => The security group to be associated with the instance.
- image-id => The AMI id by which the instance is to be launched.

- instance-type => The type of the instance.
- key-pair => The Pem file to authenticate the login process.
- count => The number of instances to be launched.
- volume-size => The size of the EBS volumes to be attached.

⇒ Step 3: *Create ansible playbook to launch EC2 instance*

Playbooks are Ansible's configuration, deployment, and orchestration language. They can describe a policy you want your remote systems to enforce, or a set of steps in a general IT process. In simple, if Ansible modules are the tools in your workshop, playbooks are your instruction manuals, and your inventory of hosts are your raw material.

Ansible uses its ec2 module to launch the instances. The module supports various parameters to customize and configure the instance according to user needs. Please access [Ansible ec2 module] to know more details about supported parameters. Here is the playbook snippet to launch ec2 instances,

```
---
- name: Configuring the EC2 Instance
  hosts: localhost
  connection: local

  tasks:
  - name: Launching an EC2 Instance
    local_action: ec2
            aws_access_key={{ aws_access_key }}
            aws_secret_key={{ aws_secret_key }}
            group={{ security_group }}
            instance_type={{ instance_type }}
            image={{ image }}
            region={{ region }}
            wait=true
            keypair={{ keypair }}
            count={{ count }}
            vpc_subnet_id={{ vpc_subnet_id }}
            assign_public_ip=yes
    register: ec2

  - name: Wait for SSH to come up
    wait_for:
      host: "{{ item.public_dns_name }}"
      port: 22
      delay: 60
      timeout: 320
      state: started
```

```
  with_items: "{{ ec2.instances }}"

- name: Add new instances to host group
  add_host: hostname={{item.public_ip}} groupname=deploy
  with_items: ec2.instances
```

Sample Ansible playbook to launch AWS EC2 instance

⇒ Step 4: *Run your play to launch EC2 instances*

Run your play using ansible-playbook command as below to launch AWS EC2 instance in your account. You may need to change the parameter values as per your requirement.

```
$ sudo ansible-playbook launch-ec2-play.yaml \
      -e instance_type="t2.micro" \
      -e keypair="ansible-key" \
      -e count=1 \
      -e image="ami-7d95b612" \
      -e security_group="default" \
      -e region="ap-south-1" \
      -e aws_access_key="XXXXXXXXXXXX" \
      -e aws_secret_key="XXXXXXXXXXXX" \
      -e vpc_subnet_id="XXXXXXXXXXXX"
```

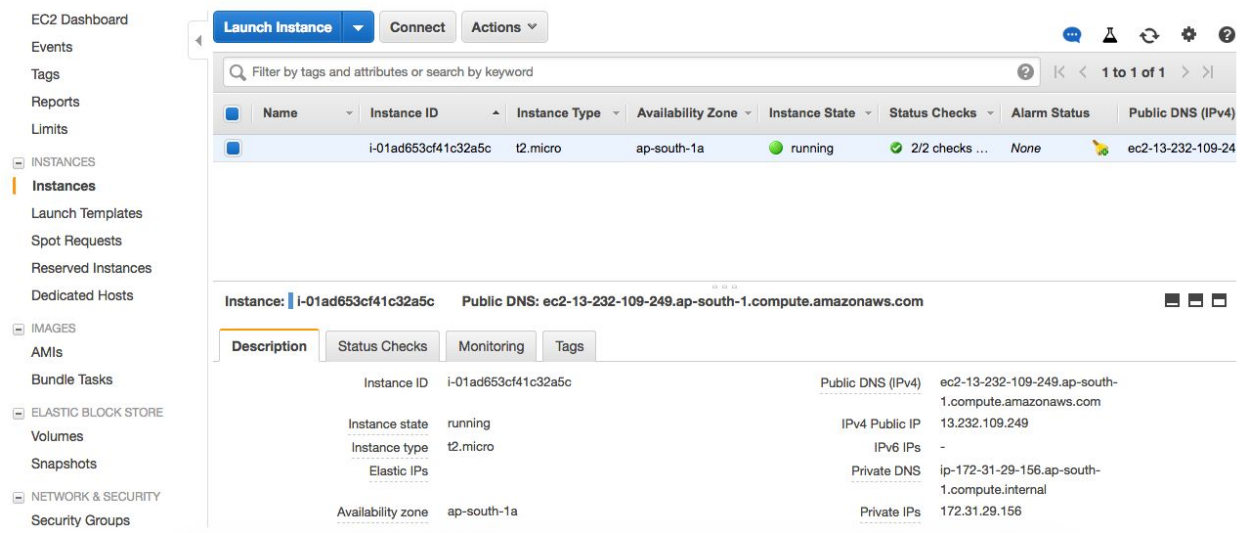
The sample output after above command look like,

```
bash-3.2$ sudo ansible-playbook launch-ec2-play.yaml -e volume-size=10 -e instance_type="t2.micro" -e keypair="ansi
ble-key" -e count=1 -e image="ami-7d95b612" -e security_group="default" -e region="ap-south-1" -e aws_access_key="[redacted]
[redacted]" -e aws_secret_key="[redacted]" -e vpc_subnet_id="subnet-4c017725
"
Password:

PLAY [Configuring the EC2 Instance] ********************************************************************************

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

TASK [Launching an EC2 Instance] ***********************************************************************************
changed: [localhost -> localhost]

TASK [Wait for SSH to come up] *************************************************************************************
ok: [localhost] => (item={u'kernel': None, u'root_device_type': u'ebs', u'private_dns_name': u'ip-172-31-20-150.ap-
south-1.compute.internal', u'public_ip': u'13.127.93.224', u'private_ip': u'172.31.20.150', u'id': u'i-0dec0a203be7
73a86', u'ebs_optimized': False, u'state': u'running', u'virtualization_type': u'hvm', u'root_device_name': u'/dev/
xvda', u'ramdisk': None, u'block_device_mapping': {u'/dev/xvda': {u'status': u'attached', u'delete_on_termination':
 True, u'volume_id': u'vol-0296d5603aa3b9005'}}, u'key_name': u'ansible-key', u'image_id': u'ami-7d95b612', u'tenan
cy': u'default', u'groups': {u'sg-d80009b1': u'default'}, u'public_dns_name': u'ec2-13-127-93-224.ap-south-1.comput
e.amazonaws.com', u'state_code': 16, u'tags': {}, u'placement': u'ap-south-1a', u'ami_launch_index': u'0', u'dns_na
me': u'ec2-13-127-93-224.ap-south-1.compute.amazonaws.com', u'region': u'ap-south-1', u'launch_time': u'2018-06-16T
20:27:06.000Z', u'instance_type': u't2.micro', u'architecture': u'x86_64', u'hypervisor': u'xen'})

PLAY RECAP *********************************************************************************************************
localhost                  : ok=3    changed=1    unreachable=0    failed=0

bash-3.2$
```

Note: The playbook task ‘Wait for SSH’ may fail if the TCP port 22 is not allowed in your inbound rules under security group

You may change the param values in the ansible playbook command according your requirement. For example, the ***instance_type*** can be changed to ‘*t2.medium*’ to launch medium type compute instance.

Now, you should be able to see newly launched machine under EC2 Dashboard > instances tab in AWS console,



Check SSH connectivity to newly launched ec2 instance,

- Get IPv4 Public IP from aws console
- Perform below SSH command

```
$ ssh -i <path-to-dir>/<your-key-pair.pem> ec2-user@<IPv4-Public-IP>
```

For example,

```
bash-3.2$ ssh -i ~/ansible-key.pem ec2-user@13.232.109.249
The authenticity of host '13.232.109.249 (13.232.109.249)' can't be established.
ECDSA key fingerprint is SHA256:T23VmXskP+x4+XVFIqDhdHqACgeRHVGu4qb9fYU9HFg.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '13.232.109.249' (ECDSA) to the list of known hosts.

       __|  __|_  )
       _|  (     /   Amazon Linux 2 AMI
      ___|\___|___|

https://aws.amazon.com/amazon-linux-2/
10 package(s) needed for security, out of 29 available
Run "sudo yum update" to apply all updates.
-bash: warning: setlocale: LC_CTYPE: cannot change locale (UTF-8): No such file or directory
[ec2-user@ip-172-31-29-156 ~]$
```

⇒ Step 5: ***Create ansible playbook task to deploy Apache web server on EC2 instance***

The Apache HTTP Server, colloquially called Apache, is a free and open-source cross-platform web server, released under the terms of Apache License 2.0.

To install Apache on ec2 instance, we use Ansible's yum module. The playbook snippet with Apache web server deployment task,

```
- name: Deploy Apache web server on EC2
  hosts: deploy
  remote_user: ec2-user
  become: yes

  tasks:
    - name: Install NTP Server
      yum: pkg=ntp state=latest

    - name: Ensure NTP is up and running
      service: name=ntpd state=started

    - name: Install Apache Web Server
      yum: pkg=httpd state=latest

    - name: Ensure httpd is up and running
      service: name=httpd state=started
```

This task instruct Ansible to,

- Install ntp package
- Enable ntpd service
- Install Apache web server
- Enable httpd apache service

⇒ Step 6: ***Create single YAML for both tasks - Launch EC2 + Deploy Apache***

Combine Playbook snippets from ***Step 3*** & ***Step 5*** to launch EC2 instance and deploy Apache web server to same instance from single playbook yaml file. Here is the complete playbook yaml,

```
---
- name: Configuring the EC2 Instance
  hosts: localhost
  connection: local
```

```
  tasks:
  - name: Launching an EC2 Instance
    local_action: ec2
                aws_access_key={{ aws_access_key }}
                aws_secret_key={{ aws_secret_key }}
                group={{ security_group }}
                instance_type={{ instance_type }}
                image={{ image }}
                region={{ region }}
                wait=true
                keypair={{ keypair }}
                count={{ count }}
                vpc_subnet_id={{ vpc_subnet_id }}
                assign_public_ip=yes
    register: ec2

  - name: Wait for SSH to come up
    wait_for:
        host: "{{ item.public_dns_name }}"
        port: 22
        delay: 60
        timeout: 320
        state: started
    with_items: "{{ ec2.instances }}"

  - name: Add new instances to host group
    add_host: hostname="{{ item.public_ip }}" groupname=deploy
    with_items: "{{ ec2.instances }}"

- name: Deploy Apache web server on EC2
  hosts: deploy
  remote_user: ec2-user
  become: yes

  tasks:
    - name: Install NTP Server
      yum: pkg=ntp state=latest

    - name: Ensure NTP is up and running
      service: name=ntpd state=started

    - name: Install Apache Web Server
      yum: pkg=httpd state=latest

    - name: Ensure httpd is up and running
      service: name=httpd state=started
```

⇒ Step 7: *Run your play to launch EC2 instances and configure Apache web servers*

So, finally run your consolidated play as below to perform both launching AWS EC2 instances and configure Apache web server on all instances.

```
$ ANSIBLE_HOST_KEY_CHECKING=False ansible-playbook launch-ec2-play.yaml \
        -e volume-size=10 \
        -e instance_type="t2.micro" \
        -e keypair="ansible-key" \
        -e count=1 \
        -e image="ami-7d95b612" \
        -e security_group="default" \
        -e region="ap-south-1" \
        -e aws_access_key="XXXXXXXXXXX" \
        -e aws_secret_key="XXXXXXXXXXX" \
        -e vpc_subnet_id="XXXXXXXXXXX" \
        --private-key=<path-to-pem-file>
```

The sample output after executing above command look like,

```
bash-3.2$ ANSIBLE_HOST_KEY_CHECKING=False ansible-playbook launch-ec2-play.yaml -e volume-size=10 -e instance_type="t2.micro" -e keypair="ansible-key" -e count=1 -e image="ami-7d95b
612" -e security_group="default" -e region="ap-south-1" -e aws_access_key="[redacted]" -e aws_secret_key="[redacted]" -e vpc_subnet_id="subne
t-4c017725" --private-key=~/ansible-key.pem

PLAY [Configuring the EC2 Instance] ********************************************************************************

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

TASK [Launching an EC2 Instance] ***********************************************************************************
changed: [localhost -> localhost]

TASK [Wait for SSH to come up] *************************************************************************************
ok: [localhost] => (item={u'kernel': None, u'root_device_type': u'ebs', u'private_dns_name': u'ip-172-31-23-174.ap-south-1.compute.internal', u'public_ip': u'13.232.36.218', u'priva
te_ip': u'172.31.23.174', u'id': u'i-007029ed8e45767e6', u'ebs_optimized': False, u'state': u'running', u'virtualization_type': u'hvm', u'root_device_name': u'/dev/xvda', u'ramdisk'
: None, u'block_device_mapping': {u'/dev/xvda': {u'status': u'attached', u'delete_on_termination': True, u'volume_id': u'vol-0dc820e911c8946d9'}}, u'key_name': u'ansible-key', u'ima
ge_id': u'ami-7d95b612', u'tenancy': u'default', u'groups': {u'sg-d80009b1': u'default'}, u'public_dns_name': u'ec2-13-232-36-218.ap-south-1.compute.amazonaws.com', u'state_code': 1
6, u'tags': {}, u'placement': u'ap-south-1a', u'ami_launch_index': u'0', u'dns_name': u'ec2-13-232-36-218.ap-south-1.compute.amazonaws.com', u'region': u'ap-south-1', u'launch_time'
: u'2018-06-17T08:15:31.000Z', u'instance_type': u't2.micro', u'architecture': u'x86_64', u'hypervisor': u'xen'})

TASK [Add new instances to host group] *****************************************************************************
changed: [localhost] => (item={u'kernel': None, u'root_device_type': u'ebs', u'private_dns_name': u'ip-172-31-23-174.ap-south-1.compute.internal', u'public_ip': u'13.232.36.218', u'
private_ip': u'172.31.23.174', u'id': u'i-007029ed8e45767e6', u'ebs_optimized': False, u'state': u'running', u'virtualization_type': u'hvm', u'root_device_name': u'/dev/xvda', u'ram
disk': None, u'block_device_mapping': {u'/dev/xvda': {u'status': u'attached', u'delete_on_termination': True, u'volume_id': u'vol-0dc820e911c8946d9'}}, u'key_name': u'ansible-key',
u'image_id': u'ami-7d95b612', u'tenancy': u'default', u'groups': {u'sg-d80009b1': u'default'}, u'public_dns_name': u'ec2-13-232-36-218.ap-south-1.compute.amazonaws.com', u'state_cod
e': 16, u'tags': {}, u'placement': u'ap-south-1a', u'ami_launch_index': u'0', u'dns_name': u'ec2-13-232-36-218.ap-south-1.compute.amazonaws.com', u'region': u'ap-south-1', u'launch_
time': u'2018-06-17T08:15:31.000Z', u'instance_type': u't2.micro', u'architecture': u'x86_64', u'hypervisor': u'xen'})

PLAY [Configure instance] ******************************************************************************************

TASK [Gathering Facts] *********************************************************************************************
ok: [ec2-13-232-36-218.ap-south-1.compute.amazonaws.com]

TASK [Install NTP Server] ******************************************************************************************
changed: [ec2-13-232-36-218.ap-south-1.compute.amazonaws.com]

TASK [Ensure NTP is up and running] ********************************************************************************
changed: [ec2-13-232-36-218.ap-south-1.compute.amazonaws.com]

TASK [Install Apache Web Server] ***********************************************************************************
[changed: [ec2-13-232-36-218.ap-south-1.compute.amazonaws.com]                                                      ]
[                                                                                                                    ]
[TASK [Ensure httpd is up and running] *****************************************************************************]
[changed: [ec2-13-232-36-218.ap-south-1.compute.amazonaws.com]                                                      ]
[                                                                                                                    ]
PLAY RECAP *********************************************************************************************************
ec2-13-232-36-218.ap-south-1.compute.amazonaws.com : ok=5    changed=4    unreachable=0    failed=0
localhost                  : ok=4    changed=2    unreachable=0    failed=0
[                                                                                                                    ]
bash-3.2$
```

⇒ **Step 8:** ***Access and verify the Apache web server test page***

Wow, you have created an EC2 instance and deployed Apache web server on to it using Ansible playbook with single command!!. Now, verify and access Apache web server test page by accessing the public dns name or ip of EC2 instance on which we installed Apache.

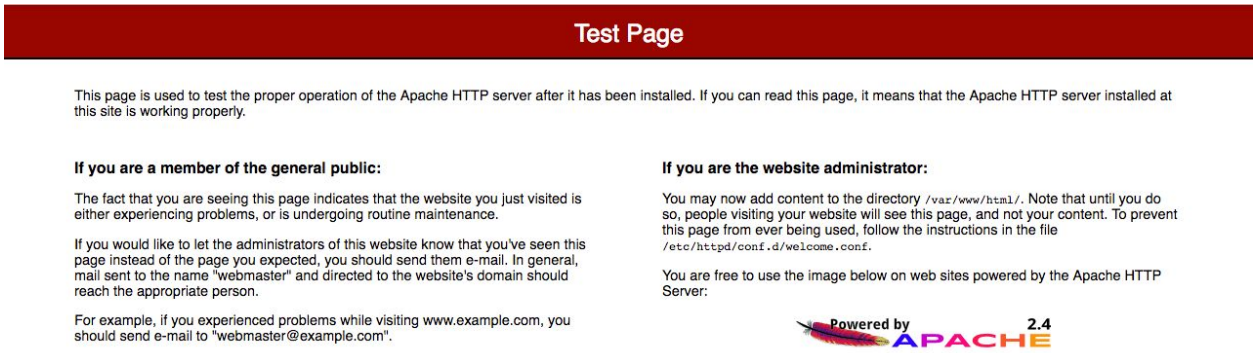
Check web server communication to newly launched ec2 instance,

- Get IPv4 Public IP or DNS name from aws console
- Access the page in your favourite browser with url, http://<ec2-instance-public-dns>,

http://ec2-13-232-36-218.ap-south-1.compute.amazonaws.com/

Note: *The inbound rule for http TCP port should be configured under security group*

You should able to see Apache test page,



Test Page

This page is used to test the proper operation of the Apache HTTP server after it has been installed. If you can read this page, it means that the Apache HTTP server installed at this site is working properly.

If you are a member of the general public:

The fact that you are seeing this page indicates that the website you just visited is either experiencing problems, or is undergoing routine maintenance.

If you would like to let the administrators of this website know that you've seen this page instead of the page you expected, you should send them e-mail. In general, mail sent to the name "webmaster" and directed to the website's domain should reach the appropriate person.

For example, if you experienced problems while visiting www.example.com, you should send e-mail to "webmaster@example.com".

If you are the website administrator:

You may now add content to the directory `/var/www/html/`. Note that until you do so, people visiting your website will see this page, and not your content. To prevent this page from ever being used, follow the instructions in the file `/etc/httpd/conf.d/welcome.conf`.

You are free to use the image below on web sites powered by the Apache HTTP Server:

Powered by APACHE 2.4

Method 2

In this case, we divide our problem into two sub-problems,

1. launching an EC2 instance
2. deploying Apache web server

We define the two subproblems as two roles under Ansible playbook and instruct each role to perform their allocated tasks.

Here is the playbook structure defined in our solution,

```
ec2-with-apache
    |
    |----  playbook.yaml
    |----  group_vars
             |---- all
    |
    |----  roles
             |
             |---- create_ec2
                      |
                      |---- tasks
                               |---- main.yaml
             |
             |---- deploy_apache
                      |
                      |---- tasks
                               |---- main.yaml
```

Now let’s define the contents of each file,

<u>Playbook</u>
(*ec2-with-apache/playbook.yaml*)

Ansible playbooks are written in the YAML data serialization format. If you don't know what a data serialization format is, think of it as a way to translate a programmatic data structure (lists, arrays, dictionaries, etc) into a format that can be easily stored to disk. The file can then be used to recreate the structure at a later point.

Each playbook contains one or more plays, which map hosts to a certain function. Ansible does this through something called tasks, which are basically module calls.

```yaml
# Configure EC2 Instances and deploy Apache

- name: provision instance for Apache
  hosts: localhost
  connection: local
  remote_user: ec2-user
  gather_facts: false
  roles:
    - create_ec2

- name: Install Apache
  hosts: deploy
  remote_user: ec2-user
  become: yes
  roles:
    - deploy_apache
```

playbook.yaml

Variables

(*ec2-with-apache/group_vars/all*)

We define all the variables required by roles and tasks in this file.

```yaml
---
# Variables here are applicable to all host groups

# AWS specific variables
aws_access_key:
aws_secret_key:
keypair:
vpc_subnet_id:
region:
image: ami-7d95b612
instance_type: t2.micro
security_group: default
instance_count: 1
wait_for_port: 22
```

group_vars/all

Note: All undefined variables in above file are mandatory parameters to be passed while executing play

Role 1 - create_ec2
(*ec2-with-apache/roles/create_ec2*)

Role is a set of tasks and additional files to configure host to serve for a certain *role*.

This role creates an EC2 instance in your AWS account and add instance(s) to ***'deploy'*** host group.

```
---
# This role launches AWS EC2 instances

- name: Launching an EC2 Instance
  local_action: ec2
            aws_access_key={{ aws_access_key }}
            aws_secret_key={{ aws_secret_key }}
            group={{ security_group }}
            instance_type={{ instance_type }}
            image={{ image }}
            region={{ region }}
            wait=true
            keypair={{ keypair }}
            count={{ instance_count }}
            vpc_subnet_id={{ vpc_subnet_id }}
            assign_public_ip=yes
  register: ec2

- name: Wait for SSH to come up
  wait_for:
      host: "{{ item.public_dns_name }}"
      port: 22
      delay: 60
      timeout: 320
      state: started
  with_items: "{{ ec2.instances }}"

- name: Add new instances to host group
  add_host: hostname="{{ item.public_ip }}" groupname=deploy
  with_items: "{{ ec2.instances }}"
```

roles/create_ec2/tasks/main.yaml

Role 2 - deploy_apache
(*ec2-with-apache/roles/deploy_apache*)

Apache web server is installed to EC2 instances using the below role,

```
---
# This role deploys Apache Http server to insatnces

- name: Install NTP Server
  yum: pkg=ntp state=latest

- name: Ensure NTP is up and running
  service: name=ntpd state=started

- name: Install Apache Web Server
  yum: pkg=httpd state=latest

- name: Ensure httpd is up and running
  service: name=httpd state=started
```

roles/deploy_apache/tasks/main.yaml

Now run you play from *'ec2-with-apache'* using ansible-playbook command after keeping all the files to appropriate locations as defined in the playbook structure.

```
$ ANSIBLE_HOST_KEY_CHECKING=False ansible-playbook playbook.yaml \
        -e keypair="XXXXXX" \
        -e region="us-west-2" \
        -e aws_access_key="XXXXXXXXXXX" \
        -e aws_secret_key="XXXXXXXXXXX" \
        -e vpc_subnet_id="XXXXXXXXXXX" \
        --private-key=<path-to-pem-file>
```

You may pass other parameters defined the vars file (group_vars/all) as per your requirement.

The example output of above command looks like,

```
[redacted] ec2-with-apache [redacted] ANSIBLE_HOST_KEY_CHECKING=False ansible-playbook playbook.yaml        -e keypair="ansible-key"        -e aws_access_key=[redacted]
 -e aws_secret_key="[redacted]        -e vpc_subnet_id="[redacted]        -e region=ap-south-1        --private-key=~/ansible-key.pem

PLAY [provision instance for Apache] ***********************************************************************************

TASK [create_ec2 : Launching an Instance] ******************************************************************************
changed: [localhost -> localhost]

TASK [create_ec2 : Wait for SSH to come up] ****************************************************************************
ok: [localhost] => (item={u'kernel': None, u'root_device_type': u'ebs', u'private_dns_name': u'ip-172-31-8-150.ap-south-1.compute.internal', u'public_ip': u'13.232.6.101', u'private
_ip': u'172.31.8.150', u'id': u'i-088b1f2ac01e5e42a', u'ebs_optimized': False, u'state': u'running', u'virtualization_type': u'hvm', u'root_device_name': u'/dev/xvda', u'ramdisk': N
one, u'block_device_mapping': {u'/dev/xvda': {u'status': u'attached', u'delete_on_termination': True, u'volume_id': u'vol-066e95567c8db3351'}}, u'key_name': u'ansible-key', u'image_
id': u'ami-7d95b612', u'tenancy': u'default', u'groups': {u'sg-d80009b1': u'default'}, u'public_dns_name': u'ec2-13-232-6-101.ap-south-1.compute.amazonaws.com', u'state_code': 16, u
'tags': {}, u'placement': u'ap-south-1b', u'ami_launch_index': u'0', u'dns_name': u'ec2-13-232-6-101.ap-south-1.compute.amazonaws.com', u'region': u'ap-south-1', u'launch_time': u'2
018-06-19T19:17:16.000Z', u'instance_type': u't2.micro', u'architecture': u'x86_64', u'hypervisor': u'xen'})

TASK [create_ec2 : Add new instances to host group] ********************************************************************
changed: [localhost] => (item={u'kernel': None, u'root_device_type': u'ebs', u'private_dns_name': u'ip-172-31-8-150.ap-south-1.compute.internal', u'public_ip': u'13.232.6.101', u'pr
ivate_ip': u'172.31.8.150', u'id': u'i-088b1f2ac01e5e42a', u'ebs_optimized': False, u'state': u'running', u'virtualization_type': u'hvm', u'root_device_name': u'/dev/xvda', u'ramdis
k': None, u'block_device_mapping': {u'/dev/xvda': {u'status': u'attached', u'delete_on_termination': True, u'volume_id': u'vol-066e95567c8db3351'}}, u'key_name': u'ansible-key', u'i
mage_id': u'ami-7d95b612', u'tenancy': u'default', u'groups': {u'sg-d80009b1': u'default'}, u'public_dns_name': u'ec2-13-232-6-101.ap-south-1.compute.amazonaws.com', u'state_code':
16, u'tags': {}, u'placement': u'ap-south-1b', u'ami_launch_index': u'0', u'dns_name': u'ec2-13-232-6-101.ap-south-1.compute.amazonaws.com', u'region': u'ap-south-1', u'launch_time'
: u'2018-06-19T19:17:16.000Z', u'instance_type': u't2.micro', u'architecture': u'x86_64', u'hypervisor': u'xen'})

PLAY [Install Apache] **************************************************************************************************

TASK [Gathering Facts] *************************************************************************************************
ok: [13.232.6.101]

TASK [deploy_apache : Install NTP Server] ******************************************************************************
changed: [13.232.6.101]

TASK [deploy_apache : Ensure NTP is up and running] ********************************************************************
changed: [13.232.6.101]

TASK [deploy_apache : Install Apache Web Server] ***********************************************************************
changed: [13.232.6.101]

TASK [deploy_apache : Ensure httpd is up and running] ******************************************************************
changed: [13.232.6.101]

PLAY RECAP *************************************************************************************************************
13.232.6.101               : ok=5    changed=4    unreachable=0    failed=0
localhost                  : ok=3    changed=2    unreachable=0    failed=0
```

Check web server communication to newly launched ec2 instance,

- Get IPv4 Public IP or DNS name from aws console
- Access the page in your favourite browser with url, http://<ec2-instance-public-dns>,

http://ec2-13-232-36-218.ap-south-1.compute.amazonaws.com/

Note: *The inbound rule for http TCP port should be configured under security group*