Core Assignment 2: Inventory file syntax

Muhannad Sinan

- Create an inventory file and learn how the servers are added.
- Inventory file has `ini` file syntax.
- Create a file with name `hosts-inv1`
  `vim hosts-inv1`
- Add three web servers and one db server. The web servers are Linux, but the db server is Windows.
- Add additional parameters in each line to add alias, `ansible_connection`, `ansible_user` and password.
- Use the below table for information about credentials.

Alias	Host	Connection	User	Password
`web1`	server1.company.com	SSH	`root`	`Pass@123`
`web2`	server2.company.com	SSH	`root`	`Pass@234`
`web3`	server3.company.com	SSH	`Root`	`pass`
`db1`	sever4.company.com	Windows	`Admin`	`Password@123`

- Also, group all the webservers under `[web_server]` and db under `[db_servers]`
- Note: For Linux use `ansible_ssh_pass` and for Windows use `ansible_password`. Connector for windows is `winrm`
- Create another group with name `all_servers` which has both `web_servers` and `db_serevrs`

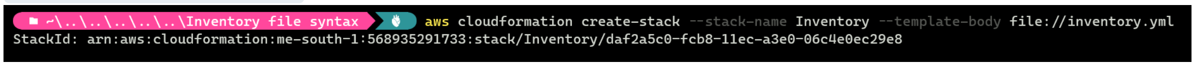
Step1:

Creating five EC2 instances, one Linux server as an Ansible master, three Linux web servers and one Windows db server using this CloudFormation template:

`inventory.yml`

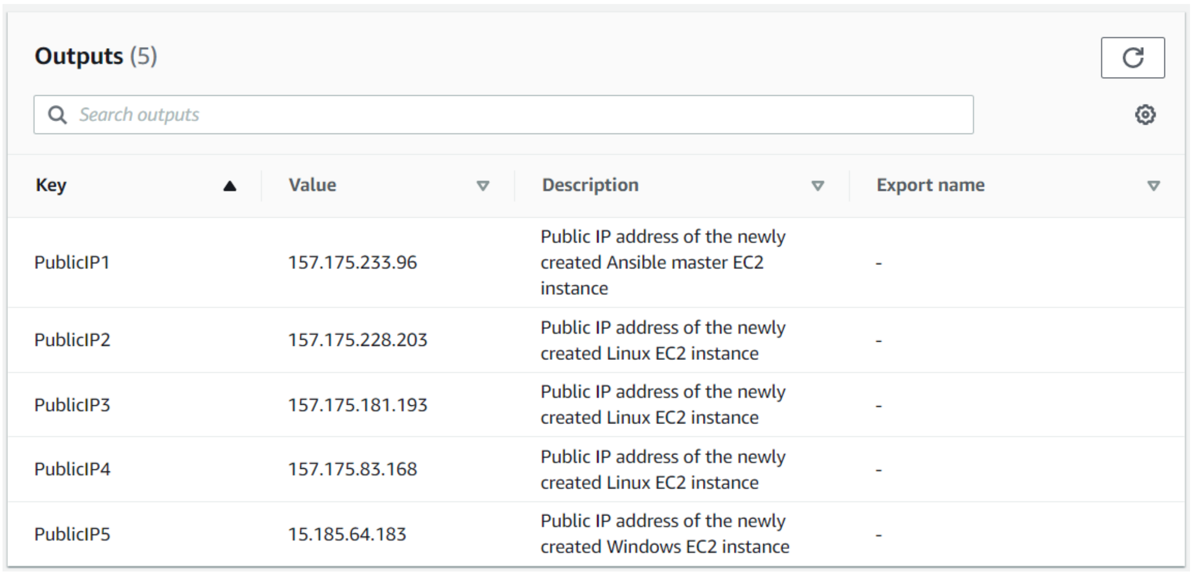
```
AWSTemplateFormatVersion: "2010-09-09"
Metadata:
  License: Apache-2.0
Description: "Core Assignment 2 - Inventory file syntax"
Parameters:
  KeyName:
    Description: Name of an existing EC2 KeyPair to enable SSH access to the
instance
    Type: AWS::EC2::KeyPair::KeyName
    Default: main
    ConstraintDescription: must be the name of an existing EC2 KeyPair.
  InstanceType:
    Description: WebServer EC2 instance type
    Type: String
    Default: t3.micro
    AllowedValues: [t3.nano, t3.micro, t3.small, t3.medium]
    ConstraintDescription: must be a valid EC2 instance type.
  SSHLocation:
    Description: The IP address range that can be used to SSH to the EC2 instances
    Type: String
    MinLength: 9
    MaxLength: 18
    Default: 0.0.0.0/0
    AllowedPattern: (\d{1,3})\.(\d{1,3})\.(\d{1,3})\.(\d{1,3})/(\d{1,2})
    ConstraintDescription: must be a valid IP CIDR range of the form x.x.x.x/x.
  HTTPLocation:
    Description: The IP address range that can be used to SSH to the EC2 instances
    Type: String
    MinLength: 9
    MaxLength: 18
    Default: 0.0.0.0/0
    AllowedPattern: (\d{1,3})\.(\d{1,3})\.(\d{1,3})\.(\d{1,3})/(\d{1,2})
    ConstraintDescription: must be a valid IP CIDR range of the form x.x.x.x/x.
  LatestAmzLinuxAmiId:
    Type: "AWS::SSM::Parameter::Value<AWS::EC2::Image::Id>"
    Default: "/aws/service/ami-amazon-linux-latest/amzn2-ami-hvm-x86_64-gp2"
  LatestWindowsAmiId:
    Type: "AWS::SSM::Parameter::Value<AWS::EC2::Image::Id>"
    Default: "/aws/service/ami-windows-latest/Windows_Server-2016-English-Full-
Base"
  InstanceCount:
    Description: Number of EC2 instances (must be between 1 and 5).
    Type: Number
    Default: 1
    MinValue: 1
    MaxValue: 5
    ConstraintDescription: Must be a number between 1 and 5.

Resources:
  MasterSecurityGroup:
    Type: AWS::EC2::SecurityGroup
    Properties:
      GroupName: ansbile-master-sg
      GroupDescription: Enable SSH access via port 22
      SecurityGroupIngress:
        - IpProtocol: tcp
          FromPort: 22
          ToPort: 22
          CidrIp: !Ref "SSHLocation"

  NodeSecurityGroup:
    Type: AWS::EC2::SecurityGroup
    Properties:
      GroupName: ansbile-node-sg
      GroupDescription: Enable SSH access via port 22 and HTTP access via port 80
      SecurityGroupIngress:
        - IpProtocol: tcp
          FromPort: 22
          ToPort: 22
          CidrIp: !Ref "SSHLocation"
        - IpProtocol: tcp
          FromPort: 80
          ToPort: 80
          CidrIp: !Ref "HTTPLocation"

  EC2InstanceMaster:
    Type: AWS::EC2::Instance
    Properties:
      InstanceType: !Ref "InstanceType"
      SecurityGroups: [!Ref "MasterSecurityGroup"]
      KeyName: !Ref "KeyName"
      ImageId: !Ref "LatestAmzLinuxAmiId"
      UserData:
        Fn::Base64: !Sub |
          #!/bin/bash
          sudo yum update -y
          sudo amazon-linux-extras install ansible2 -y
      Tags:
        - Key: Name
          Value: ansible-master
  EC2Instanceweb1:
    Type: AWS::EC2::Instance
    Properties:
      InstanceType: !Ref "InstanceType"
      SecurityGroups: [!Ref "NodeSecurityGroup"]
      KeyName: !Ref "KeyName"
      ImageId: !Ref "LatestAmzLinuxAmiId"
      Tags:
        - Key: Name
          Value: web1
  EC2Instanceweb2:
    Type: AWS::EC2::Instance
    Properties:
      InstanceType: !Ref "InstanceType"
      SecurityGroups: [!Ref "NodeSecurityGroup"]
      KeyName: !Ref "KeyName"
      ImageId: !Ref "LatestAmzLinuxAmiId"
      Tags:
        - Key: Name
          Value: web2
  EC2Instanceweb3:
    Type: AWS::EC2::Instance
    Properties:
      InstanceType: !Ref "InstanceType"
      SecurityGroups: [!Ref "NodeSecurityGroup"]
      KeyName: !Ref "KeyName"
      ImageId: !Ref "LatestAmzLinuxAmiId"
      Tags:
        - Key: Name
          Value: web3
  EC2WindowsInstanceDb1:
    Type: AWS::EC2::Instance
    Properties:
      InstanceType: !Ref "InstanceType"
      SecurityGroups: [!Ref "NodeSecurityGroup"]
      KeyName: !Ref "KeyName"
      ImageId: !Ref "LatestWindowsAmiId"
      Tags:
        - Key: Name
          Value: db1

Outputs:
  PublicIP1:
    Description: Public IP address of the newly created Ansible master EC2 instance
    Value: !GetAtt [EC2InstanceMaster, PublicIp]

  PublicIP2:
    Description: Public IP address of the newly created Linux EC2 instance
    Value: !GetAtt [EC2Instanceweb1, PublicIp]

  PublicIP3:
    Description: Public IP address of the newly created Linux EC2 instance
    Value: !GetAtt [EC2Instanceweb2, PublicIp]

  PublicIP4:
    Description: Public IP address of the newly created Linux EC2 instance
    Value: !GetAtt [EC2Instanceweb3, PublicIp]

  PublicIP5:
    Description: Public IP address of the newly created Windows EC2 instance
    Value: !GetAtt [EC2WindowsInstanceDb1, PublicIp]
```

Creating the CloudFormation stack using aws cli:
`aws cloudformation create-stack --stack-name Inventory --template-body file://inventory.yml`



```
~\..\..\..\..\..\Inventory file syntax  aws cloudformation create-stack --stack-name Inventory --template-body file://inventory.yml
StackId: arn:aws:cloudformation:me-south-1:568935291733:stack/Inventory/daf2a5c0-fcb8-11ec-a3e0-06c4e0ec29e8
```

The outputs is:

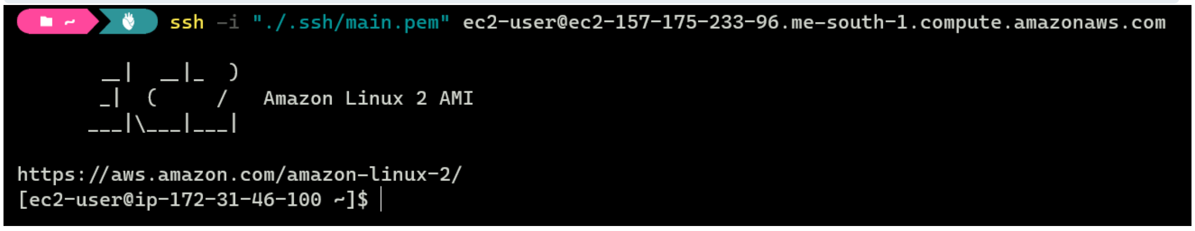


Outputs (5)

Key	Value	Description	Export name
PublicIP1	157.175.233.96	Public IP address of the newly created Ansible master EC2 instance	-
PublicIP2	157.175.228.203	Public IP address of the newly created Linux EC2 instance	-
PublicIP3	157.175.181.193	Public IP address of the newly created Linux EC2 instance	-
PublicIP4	157.175.83.168	Public IP address of the newly created Linux EC2 instance	-
PublicIP5	15.185.64.183	Public IP address of the newly created Windows EC2 instance	-

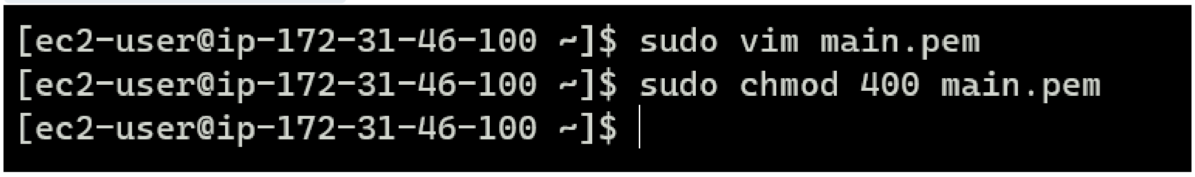
Step2:

Connect to `ansible-master` server using ssh protocol:
`ssh -i "./.ssh/main.pem" ec2-user@ec2-157-175-233-96.me-south-1.compute.amazonaws.com`



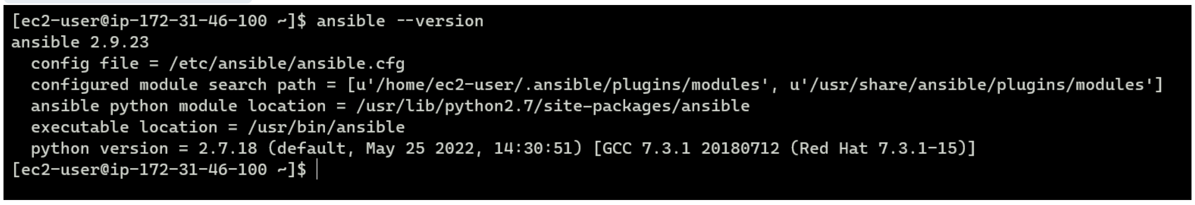
```
ssh -i "./.ssh/main.pem" ec2-user@ec2-157-175-233-96.me-south-1.compute.amazonaws.com

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

https://aws.amazon.com/amazon-linux-2/
[ec2-user@ip-172-31-46-100 ~]$
```

Add my private ssh key to the Ansible master server to this directory `/home/ec2-user/main.pem` and change the permissions to 400:
`sudo vim main.pem`
`sudo chmod 400 main.pem`



```
[ec2-user@ip-172-31-46-100 ~]$ sudo vim main.pem
[ec2-user@ip-172-31-46-100 ~]$ sudo chmod 400 main.pem
[ec2-user@ip-172-31-46-100 ~]$
```

now let's check if Ansible is installed or not:
`ansible --version`



```
[ec2-user@ip-172-31-46-100 ~]$ ansible --version
ansible 2.9.23
  config file = /etc/ansible/ansible.cfg
  configured module search path = [u'/home/ec2-user/.ansible/plugins/modules', u'/usr/share/ansible/plugins/modules']
  ansible python module location = /usr/lib/python2.7/site-packages/ansible
  executable location = /usr/bin/ansible
  python version = 2.7.18 (default, May 25 2022, 14:30:51) [GCC 7.3.1 20180712 (Red Hat 7.3.1-15)]
[ec2-user@ip-172-31-46-100 ~]$
```

Ansible version 2.9.23 is installed.

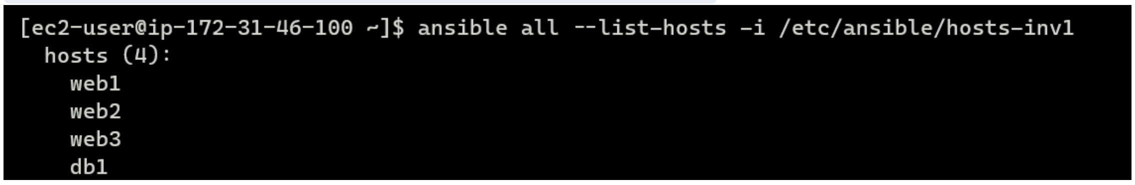
Step3:

Creating an inventory file in the path `/etc/ansible/hosts-inv1`
`sudo vim /etc/ansible/hosts-inv1`
with the following content:

```
[all_servers:children]
web_servers
db_servers

[web_servers]
web1 ansible_host=157.175.228.203 ansible_connection=ssh ansible_user=root
ansible_ssh_pass=Pass@123
web2 ansible_host=157.175.181.193 ansible_connection=ssh ansible_user=root
ansible_ssh_pass=Pass@234
web3 ansible_host=157.175.83.168 ansible_connection=ssh ansible_user=Root
ansible_ssh_pass=pass

[db_servers]
db1 ansible_host=15.185.64.183 ansible_connection=winrm ansible_user=Admin
ansible_password=Password@123

[web_servers:vars]
ansible_ssh_private_key_file=/home/ec2-user/main.pem
```

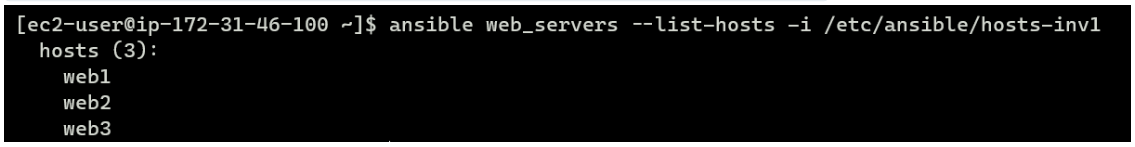
Step4:

1. Run a command to list all the hosts:
   `ansible all --list-hosts -i /etc/ansible/hosts-inv1`



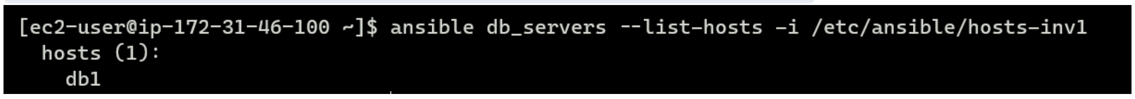
```
[ec2-user@ip-172-31-46-100 ~]$ ansible all --list-hosts -i /etc/ansible/hosts-inv1
  hosts (4):
    web1
    web2
    web3
    db1
```

2. Run a command to list down only the `web servers`:
   `ansible web_servers --list-hosts -i /etc/ansible/hosts-inv1`



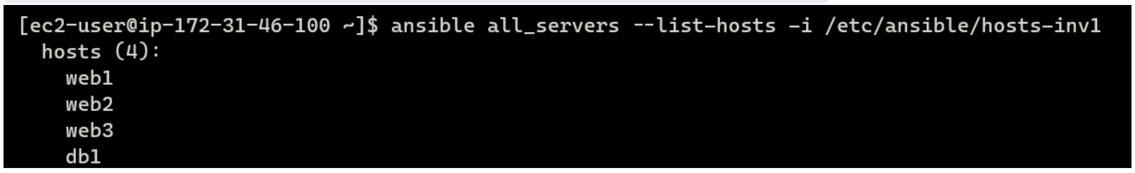
```
[ec2-user@ip-172-31-46-100 ~]$ ansible web_servers --list-hosts -i /etc/ansible/hosts-inv1
  hosts (3):
    web1
    web2
    web3
```

3. Run a command to list down only the `db servers`:
   `ansible db_servers --list-hosts -i /etc/ansible/hosts-inv1`



```
[ec2-user@ip-172-31-46-100 ~]$ ansible db_servers --list-hosts -i /etc/ansible/hosts-inv1
  hosts (1):
    db1
```

4. **Bonus:** Run a command to list down the `all_servers`:
   `ansible all_servers --list-hosts -i /etc/ansible/hosts-inv1`



```
[ec2-user@ip-172-31-46-100 ~]$ ansible all_servers --list-hosts -i /etc/ansible/hosts-inv1
  hosts (4):
    web1
    web2
    web3
    db1
```