Creating the ec2 instance with the ansible playbook

Check whether these packages are available on the local node

yum update -y

yum install python3 -y

sudo alternatives --set python /usr/bin/python3

pip3 install ansible –user

ansible –version

python --version

Step-1

Install boto3 module

Next we would need `boto3` module on the controller node as required by the ec2 module. We had already installed `boto3` when working with Ansible Facts, but you can install it using `pip3`. If you want to install these only for the current user then append `--user` to this command or use sudo as this would require root level privilege.

pip3 install boto boto3 --user

```
[devops@ip-172-31-18-61 ~]$
[devops@ip-172-31-18-61 ~]$ pip3 install boto boto3 --user
Collecting boto
```

```
[devops@ip-172-31-18-61 ~]$
[devops@ip-172-31-18-61 ~]$ pip3 install boto boto3 --user
Collecting boto
  Downloading https://files.pythonhosted.org/packages/23/10/c0b78c27298029e4454a472a1919bde20cb182dab1662cec7f2caldcc523/boto-2.49.0-py2.py3-none-any.whl (1.
4MB)
    100% |                                | 1.4MB 917kB/s
Collecting boto3
  Downloading https://files.pythonhosted.org/packages/9a/89/ff1cbb2d7b117fcfaf4ba28daa808110d51de792e9b85ab9eb32a6925ce6/boto3-1.22.7-py3-none-any.whl (132kB
)
    100% |                                | 133kB 8.4MB/s
Collecting botocore<1.26.0,>=1.25.7 (from boto3)
  Downloading https://files.pythonhosted.org/packages/bc/b3/f5fb761b52de74f122670545135ee3c24b5e8d0db3c7a4c0930a4774bb33/botocore-1.25.7-py3-none-any.whl (8.
7MB)
    100% |                                | 8.7MB 142kB/s
Collecting s3transfer<0.6.0,>=0.5.0 (from boto3)
  Downloading https://files.pythonhosted.org/packages/7b/9c/f51775ebe7df5a7aa4e7c79ed671bde94e154bd968aca8d65bb24aba0c8c/s3transfer-0.5.2-py3-none-any.whl (7
9kB)
    100% |                                | 81kB 10.4MB/s
Collecting jmespath<2.0.0,>=0.7.1 (from boto3)
  Downloading https://files.pythonhosted.org/packages/07/cb/5f001272b6faeb23c1c9e0acc04d48eaaf5c862c17709d20e3469c6e0139/jmespath-0.10.0-py2.py3-none-any.whl
Requirement already satisfied: python-dateutil<3.0.0,>=2.1 in /usr/lib/python3.6/site-packages (from botocore<1.26.0,>=1.25.7->boto3)
Collecting urllib3<1.27,>=1.25.4 (from botocore<1.26.0,>=1.25.7->boto3)
  Downloading https://files.pythonhosted.org/packages/ec/03/062e6444ce4baf1eac17a6a0ebfe36bb1ad05e1df0e20b110de59c278498/urllib3-1.26.9-py2.py3-none-any.whl
(138kB)
    100% |                                | 143kB 8.0MB/s
```

Step-2

Install awscli

We will use `awscli` to store our login credentials instead of the playbook for better security which requires `awscli` tool. We can install `awscli` using `pip3` again:

pip3 install awscli --user

```
[devops@ip-172-31-18-61 ~]$ pip3 install awscli --user
Collecting awscli
  Downloading https://files.pythonhosted.org/packages/ec/4f/b16cc262fd0ca7c9acd744331d4ce41231fa8d63456cb90b75a721a73ece/awscli-1.23.7-py3-none-any.whl (
B)
    100% |                                | 3.9MB 359kB/s
Requirement already satisfied: s3transfer<0.6.0,>=0.5.0 in ./.local/lib/python3.6/site-packages (from awscli)
Collecting docutils<0.16,>=0.10 (from awscli)
  Downloading https://files.pythonhosted.org/packages/22/cd/a6aa959dca619918ccb55023b4cb151949c64d4d5d55b3f4ffd7eee0c6e8/docutils-0.15.2-py3-none-any.whl
7kB)
    100% |                                | 552kB 2.3MB/s
Collecting colorama<0.4.5,>=0.2.5 (from awscli)
  Downloading https://files.pythonhosted.org/packages/44/98/5b86278fbbf250d239ae0ecb724f8572af1c91f4a11edf4d36a206189440/colorama-0.4.4-py2.py3-none-any.
Requirement already satisfied: botocore==1.25.7 in ./.local/lib/python3.6/site-packages (from awscli)
Collecting rsa<4.8,>=3.1.2 (from awscli)
  Downloading https://files.pythonhosted.org/packages/e9/93/0c0f002031f18b53af7a6166103c02b9c0667be528944137cc954ec921b3/rsa-4.7.2-py3-none-any.whl
```

Now we can store our access key using `awscli`. Execute `aws configure` from the console as ansible user:

aws configure

```
Successfully installed awscli-1.23.7 colorama-0.4.4 docutils-0.15.2 pyas
[devops@ip-172-31-18-61 ~]$ aws configure
AWS Access Key ID [None]: [redacted]
AWS Secret Access Key [None]: [redacted]
Default region name [None]: us-east-2
Default output format [None]:
```

ls -a

```
[devops@ip-172-31-18-61 ~]$ ls -a
.  ..  .ansible  .aws  .bash_history  .bash_logout  .bash_profile  .bashrc  .cache  .local
```

cd .aws/

```
[devops@ip-172-31-18-61 ~]$ cd .aws/
```

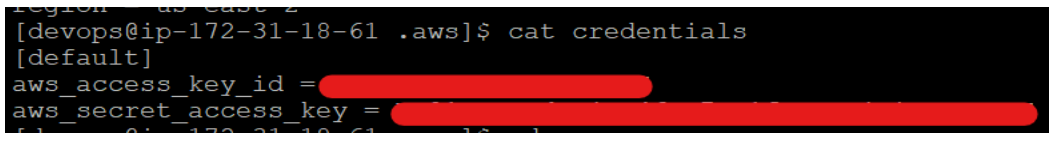
ls

```
[devops@ip-172-31-18-61 .aws]$ ls
config  credentials
```

cat config

```
[devops@ip-172-31-18-61 .aws]$   cat config
[default]
region = us-east-2
```

cat credentials



Step-3

Create ansible playbook

We are all done with the pre-requisites. Now let us create our ansible playbook to launch AWS EC2 instance using Ansible. This is our sample playbook **hello.yml** which contains multiple information about the instance. We will learn more about these in the next chapter, you can exclude those and leave everything to default if you have your custom AMI ID.

```
- name:  provisioning EC2 instances using Ansible
  hosts: localhost
  connection: local
  gather_facts: False
  tags: provisioning

  vars:
    keypair: exam
    instance_type: t2.micro
    image: ami-020db2c14939a8efb
    wait: yes
    group: good123
    count: 1
    region: us-east-1
    security_group: good123

  tasks:

    - name: Task # 1 - Create my security group
      local_action:
        module: ec2_group
        name: "{{ security_group }}"
        description: Security Group for webserver Servers
        region: "{{ region }}"
        rules:
          - proto: tcp
            from_port: 22
            to_port: 22
            cidr_ip: 0.0.0.0/0
          - proto: tcp
            from_port: 8080
            to_port: 8080
            cidr_ip: 0.0.0.0/0
          - proto: tcp
            from_port: 80
            to_port: 80
            cidr_ip: 0.0.0.0/0
        rules_egress:
          - proto: all
            cidr_ip: 0.0.0.0/0
      register: basic_firewall
    - name: Task # 2 Launch the new EC2 Instance
      local_action:  ec2
                     group={{ security_group }}
                     instance_type={{ instance_type}}
                     image={{ image }}
                     wait=true
                     region={{ region }}
                     keypair={{ keypair }}
                     count={{count}}
      register: ec2
    - name: Task # 3 Add Tagging to EC2 instance
      local_action: ec2_tag resource={{ item.id }} region={{ region }} state=present
      with_items: "{{ ec2.instances }}"
      args:
        tags:
          Name: MyTargetEc2Instance
```

Sample out

devops@ip-172-31-18-61:~

```
---
- name:  provisioning EC2 instances using Ansible
  hosts: localhost
  connection: local
  gather_facts: False
  tags: provisioning

  vars:
    keypair: exam
    instance_type: t2.micro
    image: ami-06eecef118bbf9259
    wait: yes
    group: good123
    count: 1
    region: us-east-1
    security_group: good123

  tasks:

    - name: Task # 1 - Create my security group
      local_action:
        module: ec2_group
        name: "{{ security_group }}"
        description: Security Group for webserver Servers
        region: "{{ region }}"
        rules:
           - proto: tcp
             from_port: 22
             to_port: 22
             cidr_ip: 0.0.0.0/0
           - proto: tcp
             from_port: 8080
             to_port: 8080
             cidr_ip: 0.0.0.0/0
           - proto: tcp
             from_port: 80
             to_port: 80
             cidr_ip: 0.0.0.0/0
        rules_egress:
           - proto: all
             cidr_ip: 0.0.0.0/0
      register: basic_firewall
    - name: Task # 2 Launch the new EC2 Instance
```

Step-4

Run the check

ansible-playbook hello.yml –syntax-check

```
          Name: MyTargetEc2Instance
[devops@ip-172-31-18-61 ~]$ ansible-playbook hello.yml --syntax-check
```

Sample output

```
[devops@ip-172-31-18-61 ~]$ ansible-playbook hello.yml --syntax-check
[DEPRECATION WARNING]: Ansible will require Python 3.8 or newer on the controller starting with Ansible 2.12. Current version: 3.6.8 (default, Sep  9 2021,
[redacted]

playbook: hello.yml
```

ansible-playbook hello.yml

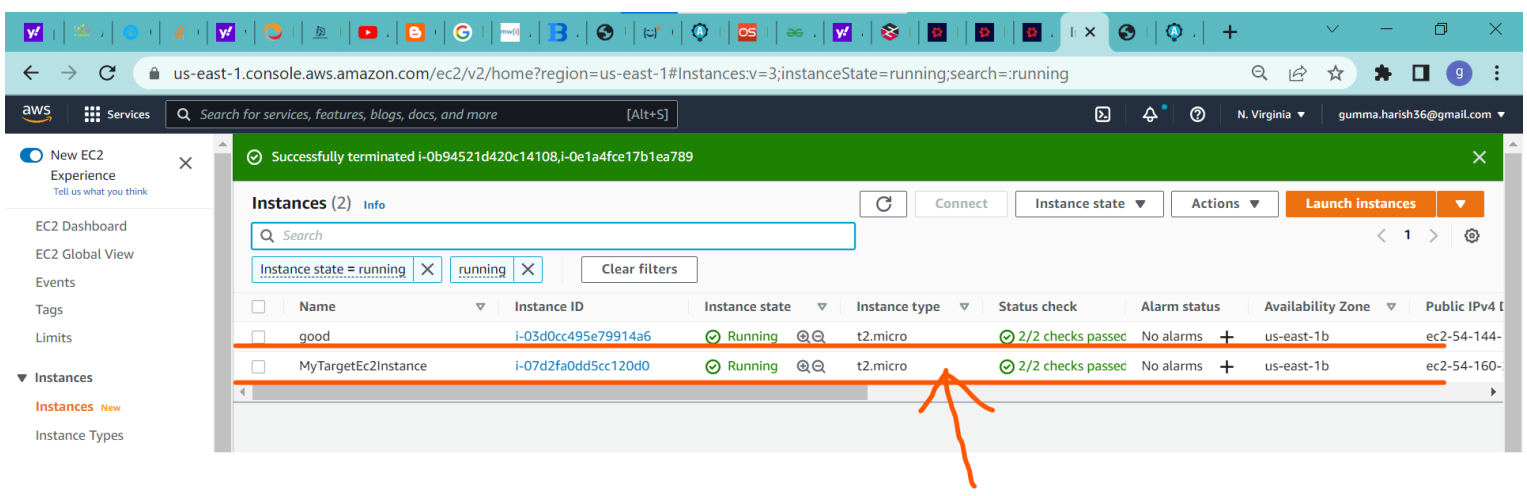
```
          Name: MyTargetEc2Instance
[devops@ip-172-31-18-61 ~]$ ansible-playbook hello.yml
```

Sample output

```
          Name: MyTargetEc2Instance
[devops@ip-172-31-18-61 ~]$ ansible-playbook hello.yml
[DEPRECATION WARNING]: Ansible will require Python 3.8 or newer on the controller starting with Ansible 2.12. Current version: 3.6.8 (default, Sep  9 2021,
07:49:02) [GCC 8.5.0 20210514 (Red Hat 8.5.0-3)]. This feature will be removed from ansible-core in version 2.12. Deprecation warnings can be disabled by
setting deprecation_warnings=False in ansible.cfg.
[WARNING]: No inventory was parsed, only implicit localhost is available
[WARNING]: provided hosts list is empty, only localhost is available. Note that the implicit localhost does not match 'all'

PLAY [provisioning EC2 instances using Ansible] ********************************************************************************************

TASK [Task] ********************************************************************************************************************************
[WARNING]: Group description does not match existing group. Descriptions cannot be changed without deleting and re-creating the security group. Try using
state=absent to delete, then rerunning this task.
ok: [localhost -> localhost]

TASK [Task] ********************************************************************************************************************************
changed: [localhost -> localhost]

TASK [Task] ********************************************************************************************************************************
changed: [localhost -> localhost] => (item={'id': 'i-0abe188ecbcbf76e6', 'ami_launch_index': '0', 'private_ip': '172.31.18.60', 'private_dns_name': 'ip-172-3
1-18-60.ec2.internal', 'public_ip': '54.174.231.36', 'dns_name': 'ec2-54-174-231-36.compute-1.amazonaws.com', 'public_dns_name': 'ec2-54-174-231-36.compute-1
.amazonaws.com', 'state_code': 16, 'architecture': 'x86_64', 'image_id': 'ami-06eecef118bbf9259', 'key_name': 'exam', 'placement': 'us-east-1b', 'region': 'u
s-east-1', 'kernel': None, 'ramdisk': None, 'launch_time': '2022-05-05T20:48:00.000Z', 'instance_type': 't2.micro', 'root_device_type': 'ebs', 'root_device_n
ame': '/dev/xvda', 'state': 'running', 'hypervisor': 'xen', 'tags': {}, 'groups': {'sg-03ce259f39c83b63e': 'good123'}, 'virtualization_type': 'hvm', 'ebs_opt
imized': False, 'block_device_mapping': {'/dev/xvda': {'status': 'attached', 'volume_id': 'vol-0efbca49f882a1729', 'delete_on_termination': True}}, 'tenancy'
: 'default'})

PLAY RECAP *********************************************************************************************************************************
localhost                  : ok=3    changed=2    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
```

Step-5

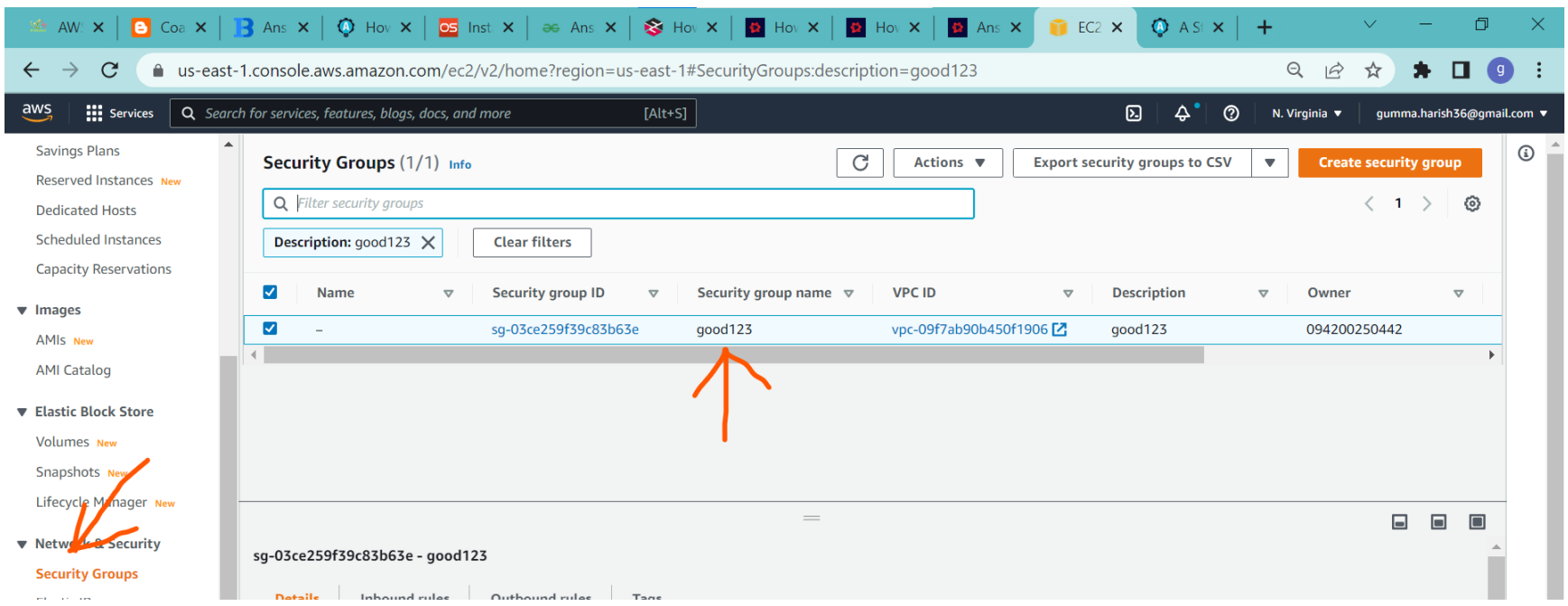
You can connect to your AWS EC2 Dashboard console and verify if a new instance has been launched:



This instance what I created through ansible playbook

Step-6

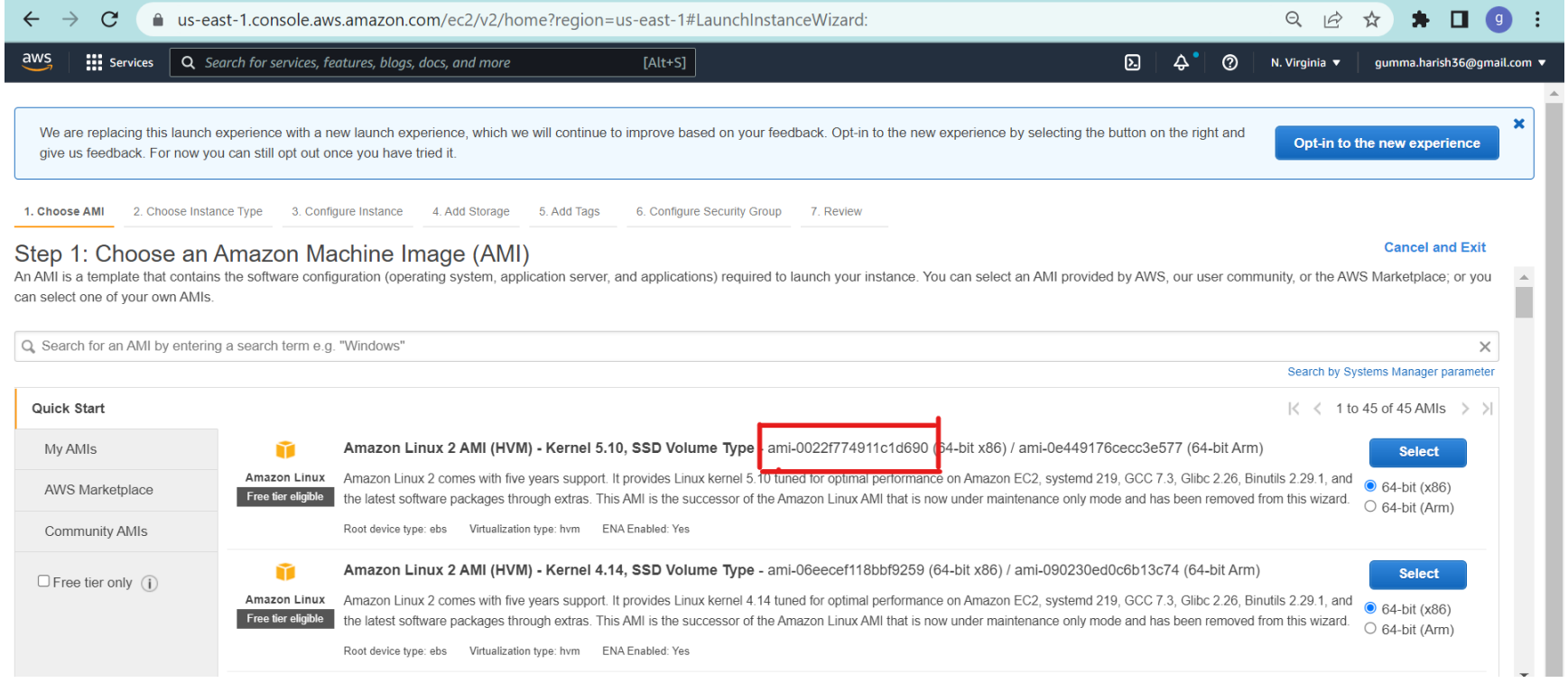
Check note (my security group name good123 .same security group I had given in the playbook)



Step-7

- Get an AWS Amazon Machine Images (AMI) ID

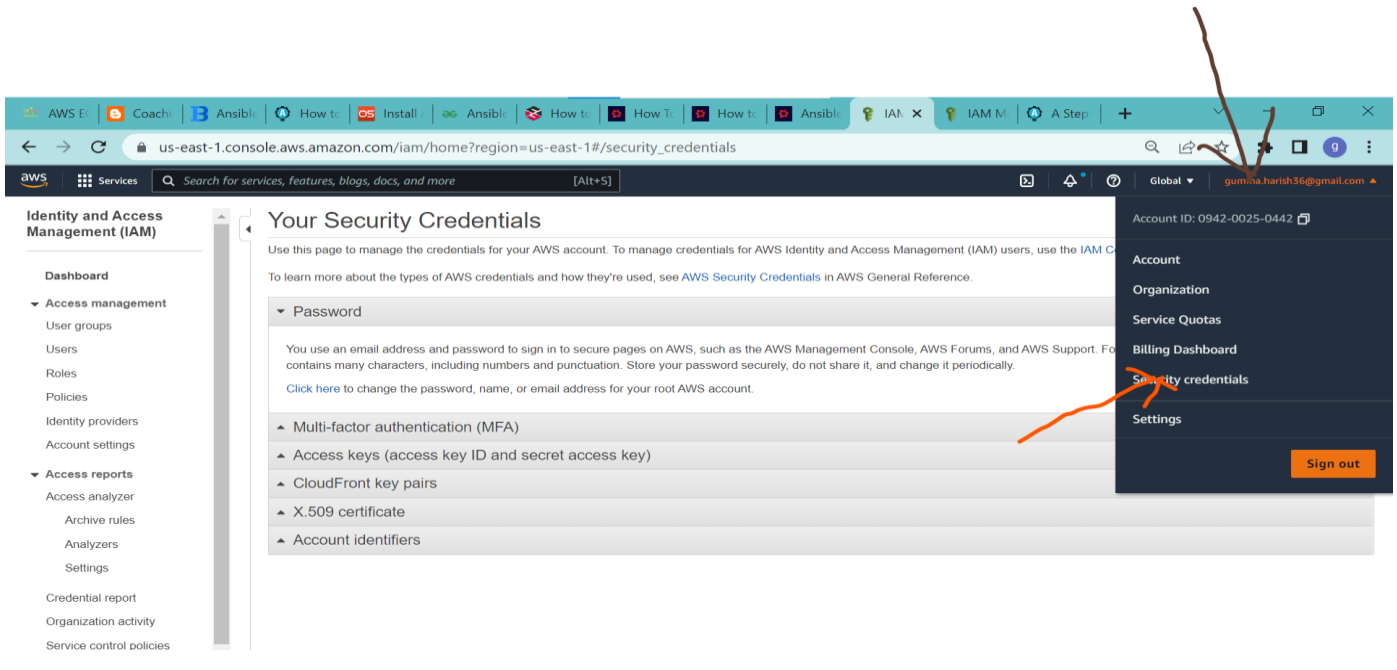
I used the that AMI ID in the playbook



Step-8

Create Access Key

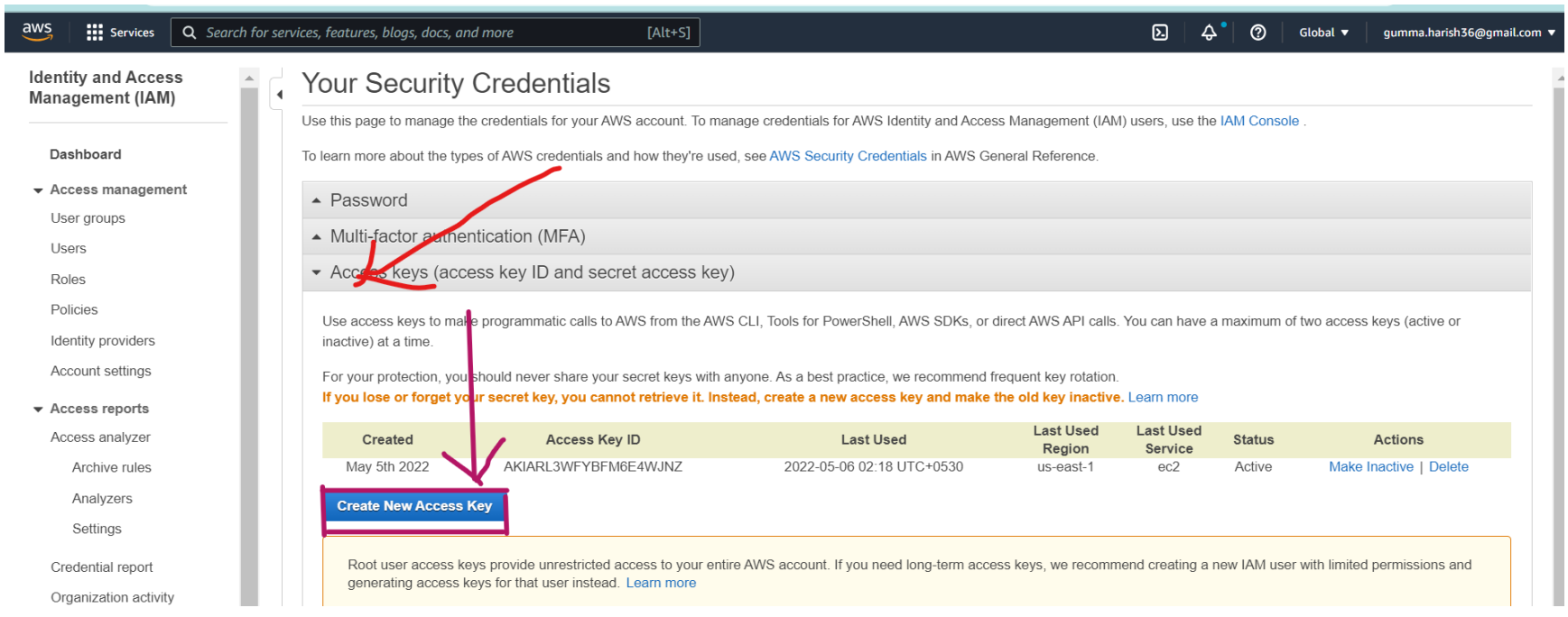
We must create an access key for our ansible on the controller node to be able to access the AWS account. Click on the **username** on your AWS Portal and from the drop down menu select "**My Security Credentials**"



1. brown colour (is the first selction option)
2. orange colour (is the second selection option)

Step-9

Now create the keys



Step-10

One more ansible palybook I executed

vim launch_ec2.yml

devops@ip-172-31-18-61:~

```
---
- name: Working with AWS EC2 Instance
  hosts: localhost
  connection: local
  gather_facts: false
  tasks:
    - name: Create ec2 instance
      ec2:
        instance_type: t2.micro
        image: ami-0022f774911c1d690
        count: 1
        key_name: exam
        group: good123
        region: us-east-1
~
~
~
```

ansible-playbook launch_ec2.yml --syntax-check

```
                        prompting)
 [devops@ip-172-31-18-61 ~]$ ansible-playbook launch_ec2.yml --syntax-check
[DEPRECATION WARNING]: Ansible will require Python 3.8 or newer on the contr
```

```
 [devops@ip-172-31-18-61 ~]$ ansible-playbook launch_ec2.yml --syntax-check
[DEPRECATION WARNING]: Ansible will require Python 3.8 or newer on the controller starting with Ansible 2.12. Current version
07:49:02) [GCC 8.5.0 20210514 (Red Hat 8.5.0-3)]. This feature will be removed from ansible-core in version 2.12. Deprecation
setting deprecation_warnings=False in ansible.cfg.
[WARNING]: No inventory was parsed, only implicit localhost is available
[WARNING]: provided hosts list is empty, only localhost is available. Note that the implicit localhost does not match 'all'

playbook: launch_ec2.yml
```

ansible-playbook launch_ec2.yml -C

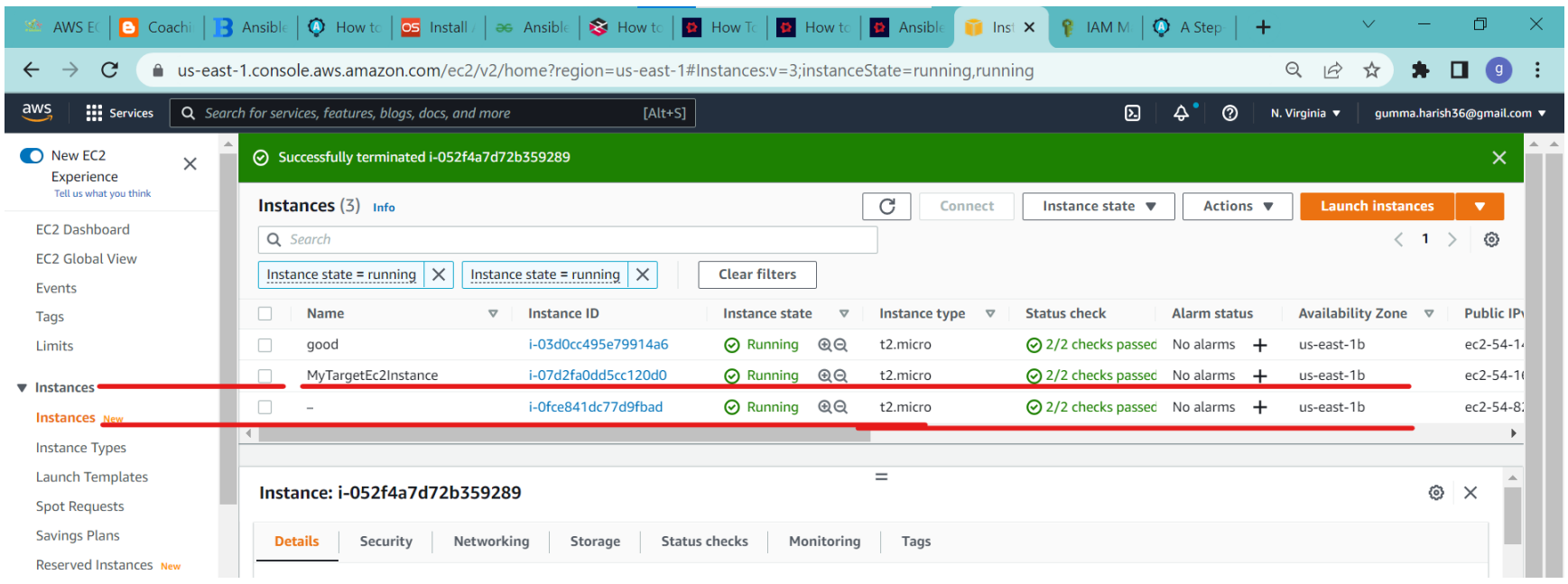
```
[devops@ip-172-31-18-61 ~]$ ansible-playbook launch_ec2.yml -C
[DEPRECATION WARNING]: Ansible will require Python 3.8 or newer on the controller starting with Ansible 2.12. Current version: 3.6.8 (default, Sep  9 2021,
07:49:02) [GCC 8.5.0 20210514 (Red Hat 8.5.0-3)]. This feature will be removed from ansible-core in version 2.12. Deprecation warnings can be disabled by
setting deprecation_warnings=False in ansible.cfg.
[WARNING]: No inventory was parsed, only implicit localhost is available
[WARNING]: provided hosts list is empty, only localhost is available. Note that the implicit localhost does not match 'all'

PLAY [Working with AWS EC2 Instance] ***********************************************************************************************************************

TASK [Create ec2 instance] *********************************************************************************************************************************
skipping: [localhost]

PLAY RECAP *************************************************************************************************************************************************
localhost                  : ok=0    changed=0    unreachable=0    failed=0    skipped=1    rescued=0    ignored=0
```

ansible-playbook launch_ec2.yml

```
[devops@ip-172-31-18-61 ~]$ ansible-playbook launch_ec2.yml
[DEPRECATION WARNING]: Ansible will require Python 3.8 or newer on the controller starting with Ansible 2.12. Current version: 3.6.8 (default, Sep  9 2021,
07:49:02) [GCC 8.5.0 20210514 (Red Hat 8.5.0-3)]. This feature will be removed from ansible-core in version 2.12. Deprecation warnings can be disabled by
setting deprecation_warnings=False in ansible.cfg.
[WARNING]: No inventory was parsed, only implicit localhost is available
[WARNING]: provided hosts list is empty, only localhost is available. Note that the implicit localhost does not match 'all'

PLAY [Working with AWS EC2 Instance] ***********************************************************************************************************************

TASK [Create ec2 instance] *********************************************************************************************************************************
changed: [localhost]

PLAY RECAP *************************************************************************************************************************************************
localhost                  : ok=1    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
```

Step-11

You can connect to your AWS EC2 Dashboard console and verify if a new instance has been launched



With same security group its has been created .sample key pair only we used