0. Infinite TOC

Friday, May 19, 2023 4:35 PM

Day 1: • Introduction to Ansible • What is Ansible and its key features • Ansible architecture • Installing Ansible • Installing Ansible on different operating systems • Configuring Ansible • Ansible Inventories • Creating inventory files • Host and group variables • Dynamic inventories • Ad-hoc Commands • Running ad-hoc commands using Ansible • Basic command syntax • Ansible modules	**Day 2:** • Ansible Playbooks • What are playbooks and why we need them • YAML syntax and structure of playbooks • Writing a basic playbook • Ansible Roles • Creating and using roles • Best practices for organizing roles • Ansible Vault • Encrypting sensitive data with Ansible Vault • Creating and managing vault files
Day 3: - <u>Additional topics to be added in the content:</u> • Jinja2 Templating • Create own custom modules • Dynamic inventories	

1. Introduction to Ansible

Friday, May 19, 2023 4:41 PM

What is Ansible and its key features?

- Ansible is an **open-source automation** tool that is used for configuration management, application deployment, and task automation.
- It is designed to be **simple, flexible, and powerful**, with a focus on ease of use and readability.

Some of the key features of Ansible include:

- **Agentless Architecture**: Ansible is an agentless tool, which means that you don't need to install any agents or software on the managed nodes. Instead, Ansible uses SSH to connect to the nodes and run commands.

- **YAML Syntax**: Ansible uses a simple and easy-to-understand YAML syntax for defining playbooks, which makes it easy for non-technical users to create and understand automation scripts.

- **Playbooks**: Ansible uses playbooks to define the automation tasks that need to be performed. Playbooks are written in YAML and can be used to perform a wide variety of tasks, including configuring servers, deploying applications, and managing infrastructure.

- **Idempotency**: Ansible is idempotent, which means that running the same playbook multiple times will always result in the same outcome, regardless of the state of the system. This makes it easy to perform automated tasks without worrying about unintended consequences.

- **Task Execution**: Ansible uses a task-based model for executing automation tasks. Each task is defined in a playbook, and Ansible executes the tasks in the order they are defined.

- **Inventory Management**: Ansible uses an inventory file to define the managed nodes and their properties. The inventory file can be in a variety of formats, including INI and YAML, and can be managed dynamically using plugins.

- **Ad-hoc Commands**: Ansible allows you to run ad-hoc commands to perform quick tasks on managed nodes without having to create a playbook. Ad-hoc commands can be run from the command line or from within a playbook.

2. Ansible architecture

Friday, May 19, 2023 4:38 PM

Ansible has a client-server architecture that is designed to be simple, flexible, and powerful.
The Ansible architecture consists of several key components, including:

- **Control Node**: The control node is where *Ansible is installed* and where the Ansible playbooks are developed and executed. It can be a physical or virtual machine, and it runs the Ansible command-line tool.

- **Managed Nodes**: The managed nodes are the *servers, network devices, or other infrastructure that Ansible manages.* Ansible uses SSH to connect to these nodes and run commands.

- **Inventory**: The inventory is a *file that contains a list of the managed nodes* and their properties. It can be in a variety of formats, including INI and YAML, and can be managed dynamically using plugins.

- **Modules**: Modules are the units of work that Ansible uses to perform tasks on the managed nodes. Ansible has a large library of modules for performing tasks such as *installing packages, managing users, and copying files.*

- **Playbooks**: Playbooks are the files that *contain the automation tasks* that Ansible performs. Playbooks are written in YAML and can be used to perform a wide variety of tasks, including configuring servers, deploying applications, and managing infrastructure.

- **Task Execution**: Ansible uses a task-based model for executing automation tasks. Each task is defined in a playbook, and Ansible *executes the tasks in the order they are defined*.

- **API**: Ansible also has an API that allows developers to integrate Ansible into their own applications or workflows.

3. Installing Ansible

Friday, May 19, 2023 4:41 PM

# Install EPEL repo (Centos/RedHat)
```
yum install epel-release
```

# Install ansible package (Centos/RedHat)
```
yum install -y ansible
```

# Upgrade & update (ubuntu)
```
apt upgrade -y && apt update -y
init 6
```

# Install the software-properties-common package (ubuntu)
```
apt install software-properties-common
```

# Install ansible personal package archive (ubuntu)
```
apt-add-repository ppa:ansible/ansible
```

# Install ansible (ubuntu)
```
apt update && apt install ansible
```

4. Creating SSH key with ED22519 algorithm

Friday, May 19, 2023 4:41 PM

# Create a new directory (ansible) and an inventory file (inventory) in it:

```
mkdir ansible
vim inventory
192.168.1.5
192.168.1.6
:wq!
```

# Creating a secured SSH key using ED25519 algo & with a comment "ansible"

```
ssh-keygen -t ed25519 -C "ansible"
```

# Copying the SSH key to Node 1

```
ssh-copy-id -i /home/jeetu/.ssh/id_ed25519.pub 192.168.1.5
```

# Copying the SSH key to Node 2

```
ssh-copy-id -i /home/jeetu/.ssh/id_ed25519.pub 192.168.1.6
```

# Verify the access (smooth)

```
ssh 192.168.1.5
ssh 192.168.1.6
```

# Checking installed ansible version

```
ansible --version
```

Getting started: https://docs.ansible.com/ansible/latest/getting_started/get_started_playbook.html#get-started-playbook

Playbook intro: https://docs.ansible.com/ansible/latest/playbook_guide/playbooks_int ro.html#playbooks-intro

Using Ubuntu:

```
root@ubuntu-vm-0:~# adduser ansibleuser
Adding user `ansibleuser' ...
Adding new group `ansibleuser' (1001) ...
Adding new user `ansibleuser' (1001) with group `ansibleuser' ...
Creating home directory `/home/ansibleuser' ...
Copying files from `/etc/skel' ...
New password:
Retype new password:
passwd: password updated successfully
Changing the user information for ansibleuser
Enter the new value, or press ENTER for the default
        Full Name []: Ansible user
        Room Number []:
        Work Phone []:
        Home Phone []:
        Other []:
Is the information correct? [Y/n] Y
root@ubuntu-vm-0:~# su - ansibleuser
ansibleuser@ubuntu-vm-0:~$
```

```
############################## ERROR ###################################
# "Server refused our key" Only from MobaXterm
# The functionality of these old keys can be restored by adding
# PubkeyAcceptedKeyTypes +ssh-rsa
# to /etc/ssh/sshd_config and restarting sshd.
```

5. Pinging servers

Friday, May 19, 2023 4:42 PM

# Pinging all hosts within inventory (created above)
# ansible -i inventory all -m ping #shortcut version

# ansible all --key-file ~/.ssh/ansible -i inventory -m ping #actual version

ls--key-file = file path for ssh private key

# **Pinging specific host (192.168.1.5)**
ansible -i inventory all -m ping --limit 192.168.1.5

6. Creating and copying a file to all remote servers

Friday, May 19, 2023 4:42 PM

# Create a new dummy file with some (rough) content in it at /tmp

```
cat > /tmp/randomFile.txt
```

# Copy this /tmp/randomFile.txt file to all the hosts within inventory file at /tmp:

```
ansible -i inventory all -m copy -a "src=/tmp/randomFile.txt dest=/tmp/randomFile.txt"
```

7. Installing finger package on ubuntu

Friday, May 19, 2023 4:42 PM

# Installing package (finger) on ubuntu server group (finger-pkg)

```
ansible -i inventory finger-pkg -m apt -a "name=finger state=present" --become --ask-become-pass
```

# Installing package (finger) on CENTOS server group (finger-pkg)

```
ansible -i inventory finger-pkg -m yum -a "name=finger state=present" --become --ask-become-pass
```

# Uninstalling package (finger) on CENTOS server group (finger-pkg)

```
ansible -i inventory finger-pkg -m yum -a "name=finger state=absent" --become --ask-become-pass
```

here:

-i	inventory file path
-m	module name
-a "name=finger state=present	passes the package name ("finger") and the desired state ("present" meaning installed) to the apt module.
--become	become sudo user
--ask-become-pass	ask sudo user password
finger-pkg	Is the group name that needs to be added in the inventory file.

8. Creating local config file for ansible command shortening

Friday, May 19, 2023 4:42 PM

Creating local config file for ansible command shortening:

```
vim ansible.cfg                    #no change in name
      [defaults]
      inventory = inventory
      private_key_file = ~/.ssh/id_ed25519
:wq!
```

Testing these short commands with the help of local config file:

```
# Tesing the short cfg file
      ansible all -m ping
      ansible all --list-hosts
      ansible all -m gather_facts                          # gathering information
      ansible all -m gather_facts --limit <IP-address>   # targeting specific server.
      ansible all -m file -a "dest=/path/to/the/location mode = 777 owner = user1 group = user1 state = directory"
```

The following command checks if yum package is installed or not, but does not update it.

```
      # ansible all -m yum -a "name = package-name state = present"
```

The following command check the package is not installed.

```
      # ansible all -m yum -a "name = package-name state = absent"
```

The following command checks the latest version of package is installed.

```
      # ansible all -m yum -a "name = package-name state = latest"
```

Running other ad-hoc commands:

```
# Ad-hoc command to reboot all servers
      ansible all -m shell -a 'sudo shutdown -r now'
```

9. Creating inventory

Friday, May 19, 2023 4:43 PM

- Ansible automates tasks on managed nodes or "hosts" in your infrastructure, using a list or group of lists known as inventory.
- The simplest inventory is a single file with a list of hosts and groups.
- The default location for this file is **/etc/ansible/hosts**.
  - You can specify a different inventory file at the command line using the -i <path> option or in configuration using inventory.
- Here are three options/formats beyond the /etc/ansible/hosts file:
  - You can create a directory with multiple inventory files. (link)
  - You can pull inventory dynamically. (link)
  - You can use multiple sources for inventory, including both dynamic inventory and static files. (link)

Basic inventory file

```
all:
  hosts:
    mail.example.com:
  children:
    webservers:
      hosts:
        foo.example.com:
        bar.example.com:
    dbservers:
      hosts:
        one.example.com:
        two.example.com:
        three.example.com:
```

Default groups

- Even if you do not define any groups in your inventory file, Ansible creates two default groups: all and ungrouped.
- The all group contains every host. The ungrouped group contains all hosts that don't have another group aside from all.

Hosts in multiple groups

```
all:
  hosts:
    mail.example.com:
  children:
    webservers:
      hosts:
        foo.example.com:
        bar.example.com:
    dbservers:
      hosts:
        one.example.com:
        two.example.com:
        three.example.com:
    east:
      hosts:
        foo.example.com:
        one.example.com:
        two.example.com:
    west:
      hosts:
        bar.example.com:
        three.example.com:
    prod:
      hosts:
        foo.example.com:
        one.example.com:
        two.example.com:
    test:
      hosts:
        bar.example.com:
        three.example.com:
```

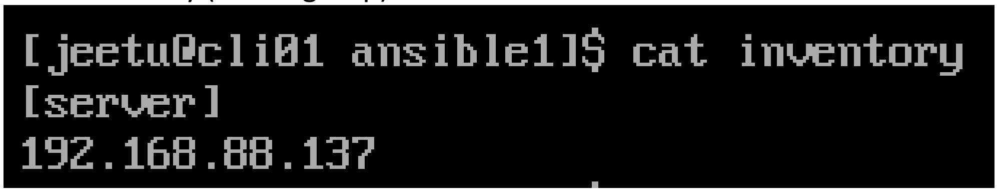
Grouping groups: parent/child group relationships

```
all:
  hosts:
    mail.example.com:
  children:
    webservers:
      hosts:
        foo.example.com:
        bar.example.com:
    dbservers:
      hosts:
        one.example.com:
        two.example.com:
        three.example.com:
    east:
      hosts:
        foo.example.com:
        one.example.com:
        two.example.com:
    west:
      hosts:
        bar.example.com:
        three.example.com:
    prod:
      children:
        east:
    test:
      children:
        west:
```

10 Playbook Scripts Demo

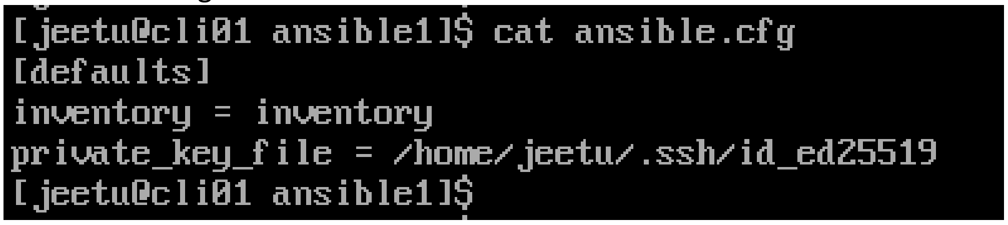
Thursday, March 14, 2024 2:01 PM

# cat inventory (with a group)



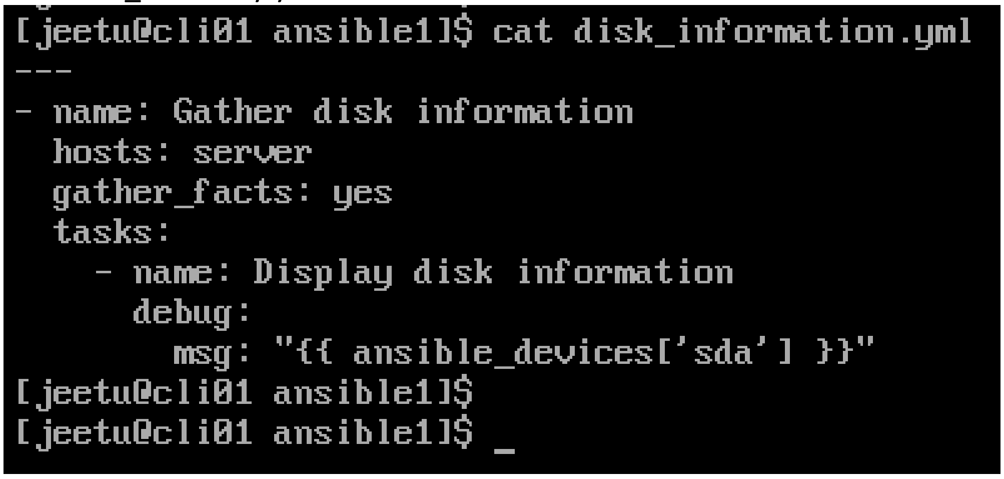
```
[jeetu@cli01 ansible1]$ cat inventory
[server]
192.168.88.137
```

Creating default config
# cat ansible.cfg



```
[jeetu@cli01 ansible1]$ cat ansible.cfg
[defaults]
inventory = inventory
private_key_file = /home/jeetu/.ssh/id_ed25519
[jeetu@cli01 ansible1]$
```

Playbook 1 : list all disk info
# cat disk_inventory.yml



```
[jeetu@cli01 ansible1]$ cat disk_information.yml
---
- name: Gather disk information
  hosts: server
  gather_facts: yes
  tasks:
    - name: Display disk information
      debug:
        msg: "{{ ansible_devices['sda'] }}"
[jeetu@cli01 ansible1]$
[jeetu@cli01 ansible1]$ _
```

Code:

```
---
- name: Gather disk information
  hosts: server
  gather_facts: yes

  tasks:
    - name: Display disk information
      debug:
        msg: "{{ ansible_devices['sda'] }}"
```

Playbook 2 : list all installed package

List all with names & versions

```
---
- name: List installed packages on CentOS
  hosts: server
  tasks:
    - name: Get installed packages
      yum:
        list: installed
      register: installed_packages

    - name: Display installed packages
      debug:
        msg: "{{ installed_packages }}"

----------------------------------
# ansible-playbook -i inventory <file-name>.yml
```

List only names

```
------------------------------------


---
- name: List installed packages on CentOS 7 and older
  hosts: server
  gather_facts: yes

  tasks:
    - name: Get installed package list
      ansible.builtin.shell: yum list installed | awk '{print $1}'
      register: installed_packages

    - name: Display installed package list
      debug:
        msg: "{{ installed_packages.stdout_lines }}"
----------------------------------
# ansible-playbook -i inventory <file-name>.yml
```

Playbook 3 : creating a user on remote machine(s)

1. Create encrypted password for the YAML.
   Open gitbash to encrypt:
   `# openssl passwd -1 -stdin <<< pass@word1`

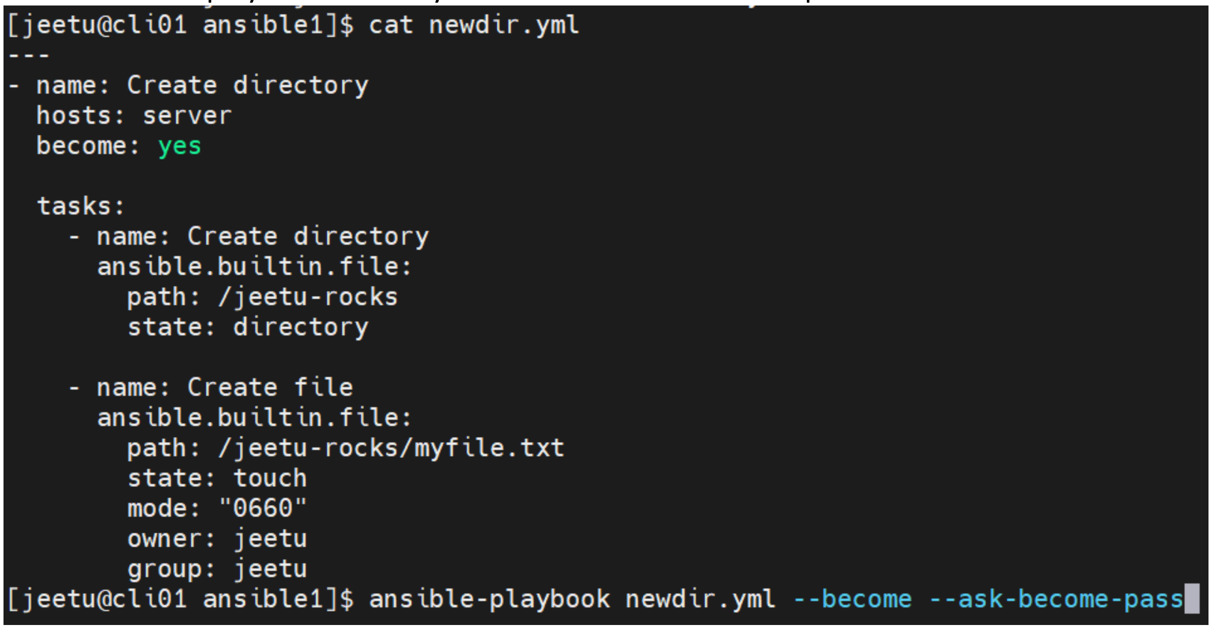
2. Cat create_user.yml

```
---
- name: Manage users
  hosts: server
  become: yes

  tasks:
    - name: Create user
      ansible.builtin.user:
        name: username
        password: <$1$cqwvuVtX$6kTCPtlzMXExirchK.YSm/>
        state: present
```

Playbook 4 : creating a directory & file with custom permissions

```
[jeetu@cli01 ansible1]$ cat newdir.yml
---
- name: Create directory
  hosts: server
  become: yes

  tasks:
    - name: Create directory
      ansible.builtin.file:
        path: /jeetu-rocks
        state: directory

    - name: Create file
      ansible.builtin.file:
        path: /jeetu-rocks/myfile.txt
        state: touch
        mode: "0660"
        owner: jeetu
        group: jeetu
```

To run: ansible-playbook newdir.yml --become --ask-become-pass



```
[jeetu@cli01 ansible1]$ cat newdir.yml
---
- name: Create directory
  hosts: server
  become: yes

  tasks:
    - name: Create directory
      ansible.builtin.file:
        path: /jeetu-rocks
        state: directory

    - name: Create file
      ansible.builtin.file:
        path: /jeetu-rocks/myfile.txt
        state: touch
        mode: "0660"
        owner: jeetu
        group: jeetu
[jeetu@cli01 ansible1]$ ansible-playbook newdir.yml --become --ask-become-pass
```

Playbook 5 : Installing packages in bulk.

```
---
- name: Install packages
  hosts: server
  become: yes

  tasks:
    - name: Install required packages
      ansible.builtin.package:
        name: "{{ item }}"
        state: present
      loop:
```

```
    - finger
    - squid
    - httpd
```

To run: ansible-playbook install_bulk_pkgs.yml --become --ask-become-pass

Playbook 6 : Copying file from local to remote

```
---
- name: Copy files
  hosts: server
  become: yes

  tasks:
    - name: Copy files from local to remote
      ansible.builtin.copy:
        src: /README.txt
        dest: /README.txt
```

Playbook 7 : Copying file from remote to local

```
---
- name: Copy file from remote to local
  hosts: all
  gather_facts: no

  tasks:
    - name: Fetch file from remote machine
      ansible.builtin.fetch:
        src: /home/jeetu/remote.txt
        dest: /home/jeetu/remote.txt
        flat: yes
```

Script (GitHub): https://github.com/jitendrastomar5593/ansible-scripts

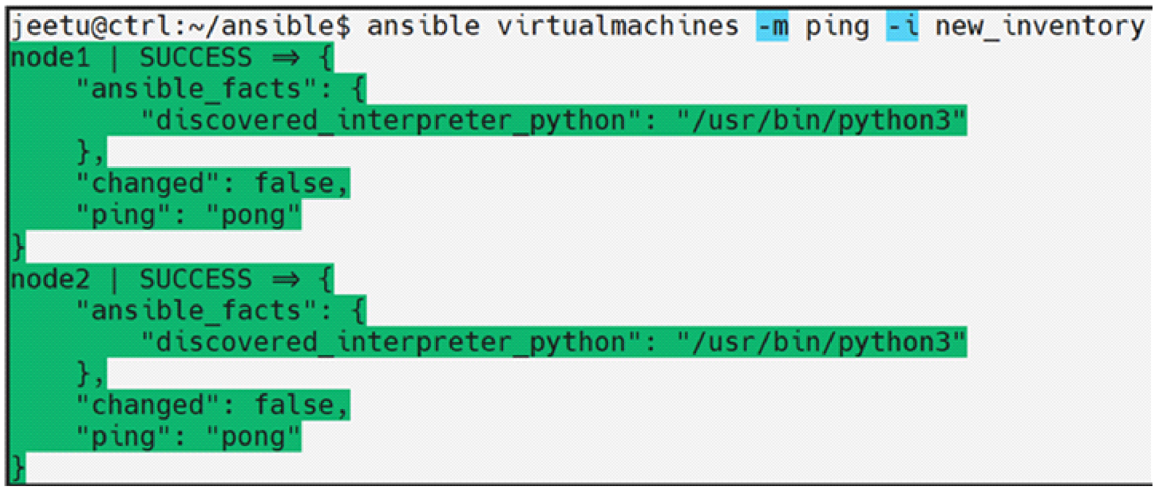
11. Creating inventory file in a different way

Friday, May 19, 2023 4:43 PM

METHOD - 1:

```
jeetu@ctrl:~/ansible$ vim new_inventory
       virtualmachines:
         hosts:
           node1:
             ansible_host: 192.168.1.5
           node2:
             ansible_host: 192.168.1.6
       :wq!
```

Output:



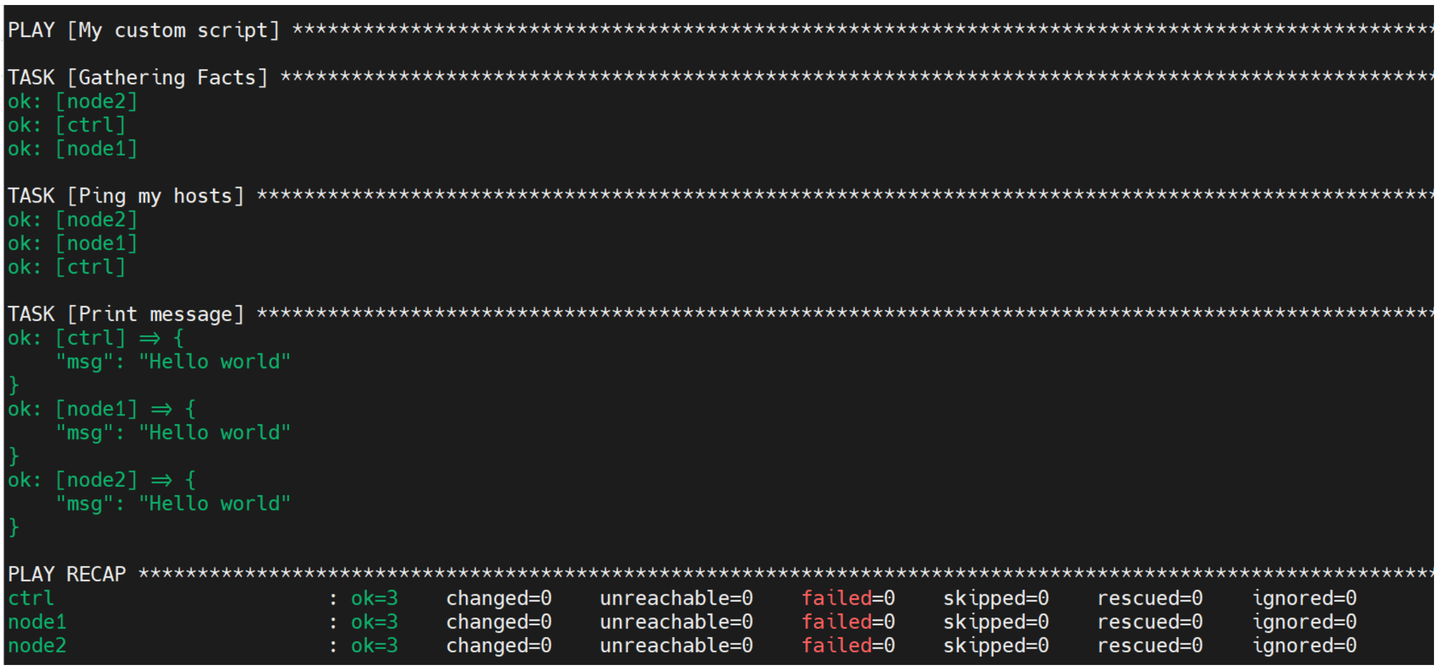
```
jeetu@ctrl:~/ansible$ ansible virtualmachines -m ping -i new_inventory
node1 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
node2 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
```

METHOD - 2:

```
jeetu@ctrl:~/ansible$ cat playbook.yml
- name: My custom script
  hosts: ubuntu
  tasks:
   - name: Ping my hosts
     ansible.builtin.ping:
   - name: Print message
     ansible.builtin.debug:
       msg: Hello world
```

Output:

```
# ansible-playbook -i new_inventory playbook.yml
```



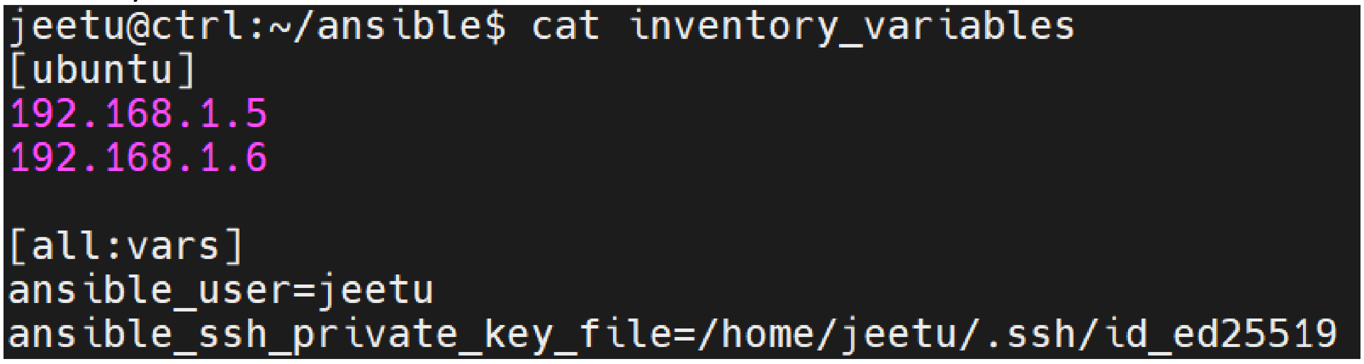
```
PLAY [My custom script] ************************************************************************************************

TASK [Gathering Facts] *************************************************************************************************
ok: [node2]
ok: [ctrl]
ok: [node1]

TASK [Ping my hosts] ***************************************************************************************************
ok: [node2]
ok: [node1]
ok: [ctrl]

TASK [Print message] ***************************************************************************************************
ok: [ctrl] => {
    "msg": "Hello world"
}
ok: [node1] => {
    "msg": "Hello world"
}
ok: [node2] => {
    "msg": "Hello world"
}

PLAY RECAP *************************************************************************************************************
ctrl                       : ok=3    changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
node1                      : ok=3    changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
node2                      : ok=3    changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
```

12. Variables in ansible inventory

Friday, May 19, 2023 4:43 PM

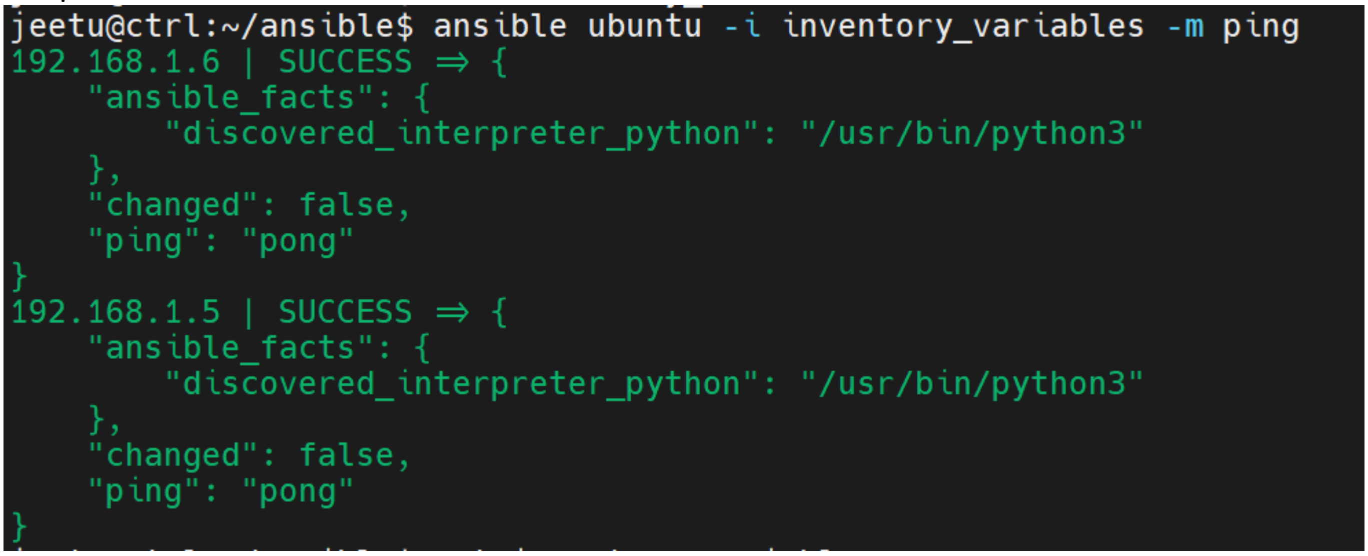
In Ansible, an inventory is a file containing a list of target hosts, grouped into different categories or variables. Variables can be used to define host-specific configuration settings, group-specific configurations, or global configurations that apply to all hosts.

- Inventory file with variables in it.



```
jeetu@ctrl:~/ansible$ cat inventory_variables
[ubuntu]
192.168.1.5
192.168.1.6

[all:vars]
ansible_user=jeetu
ansible_ssh_private_key_file=/home/jeetu/.ssh/id_ed25519
```

- Output:



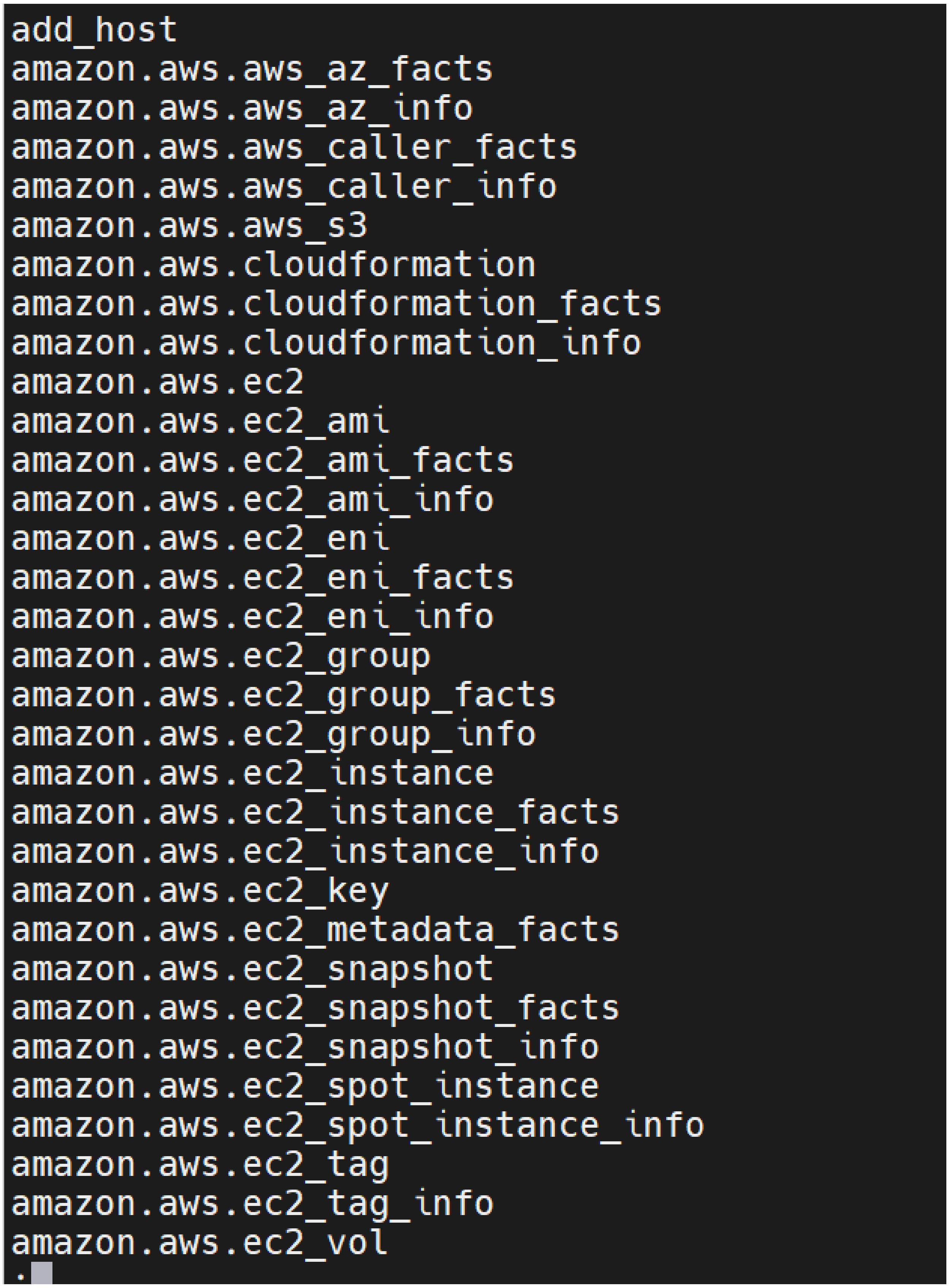
```
jeetu@ctrl:~/ansible$ ansible ubuntu -i inventory_variables -m ping
192.168.1.6 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
192.168.1.5 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
```

13. Ansible Modules

Friday, May 19, 2023 4:43 PM

Listing all modules in ansible:

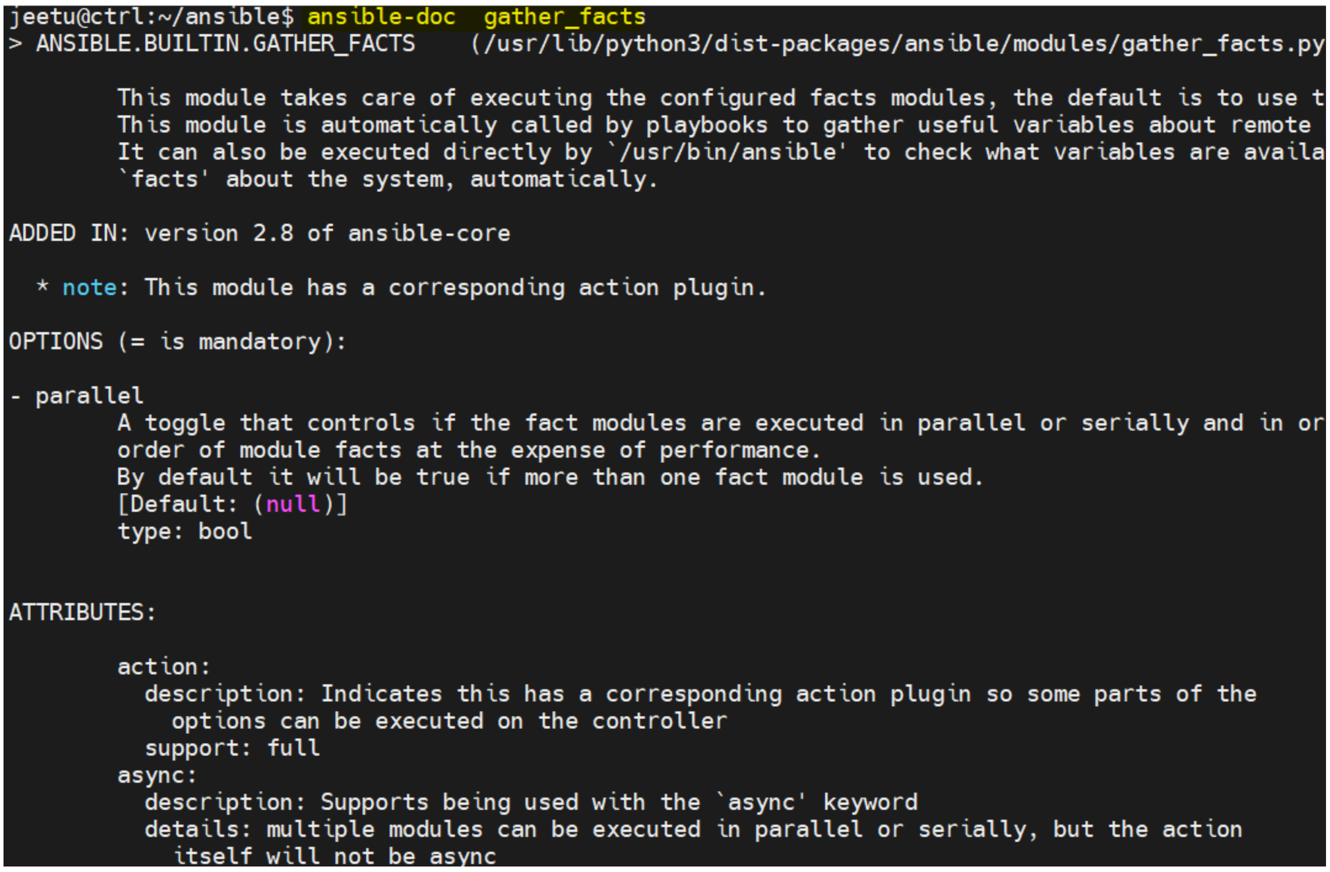
- ansible-doc --list



```
add_host
amazon.aws.aws_az_facts
amazon.aws.aws_az_info
amazon.aws.aws_caller_facts
amazon.aws.aws_caller_info
amazon.aws.aws_s3
amazon.aws.cloudformation
amazon.aws.cloudformation_facts
amazon.aws.cloudformation_info
amazon.aws.ec2
amazon.aws.ec2_ami
amazon.aws.ec2_ami_facts
amazon.aws.ec2_ami_info
amazon.aws.ec2_eni
amazon.aws.ec2_eni_facts
amazon.aws.ec2_eni_info
amazon.aws.ec2_group
amazon.aws.ec2_group_facts
amazon.aws.ec2_group_info
amazon.aws.ec2_instance
amazon.aws.ec2_instance_facts
amazon.aws.ec2_instance_info
amazon.aws.ec2_key
amazon.aws.ec2_metadata_facts
amazon.aws.ec2_snapshot
amazon.aws.ec2_snapshot_facts
amazon.aws.ec2_snapshot_info
amazon.aws.ec2_spot_instance
amazon.aws.ec2_spot_instance_info
amazon.aws.ec2_tag
amazon.aws.ec2_tag_info
amazon.aws.ec2_vol
:
```

Listing details about specific module:

- ansible-doc <module_name>
- ansible-doc gather_facts
- ansible-doc ping



```
jeetu@ctrl:~/ansible$ ansible-doc  gather_facts
> ANSIBLE.BUILTIN.GATHER_FACTS    (/usr/lib/python3/dist-packages/ansible/modules/gather_facts.py

        This module takes care of executing the configured facts modules, the default is to use t
        This module is automatically called by playbooks to gather useful variables about remote
        It can also be executed directly by `/usr/bin/ansible' to check what variables are availa
        `facts' about the system, automatically.

ADDED IN: version 2.8 of ansible-core

  * note: This module has a corresponding action plugin.

OPTIONS (= is mandatory):

- parallel
        A toggle that controls if the fact modules are executed in parallel or serially and in or
        order of module facts at the expense of performance.
        By default it will be true if more than one fact module is used.
        [Default: (null)]
        type: bool


ATTRIBUTES:

        action:
          description: Indicates this has a corresponding action plugin so some parts of the
            options can be executed on the controller
          support: full
        async:
          description: Supports being used with the `async' keyword
          details: multiple modules can be executed in parallel or serially, but the action
            itself will not be async
```

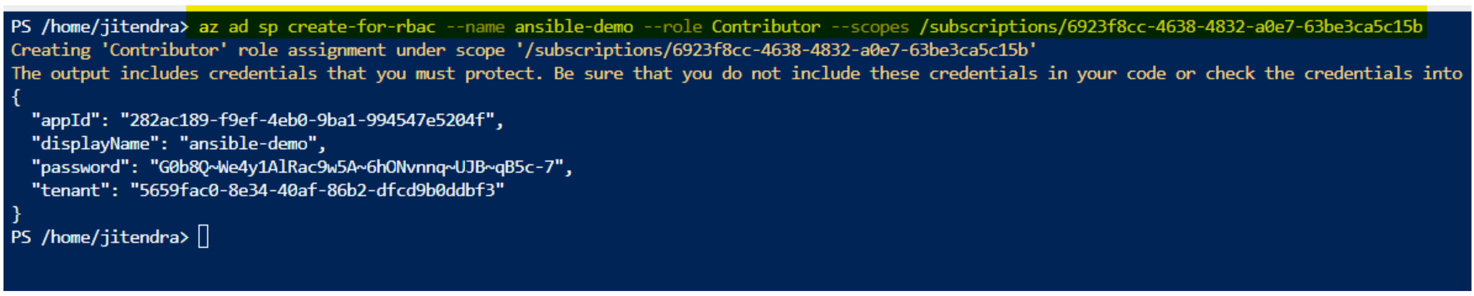
14. Dynamic inventory with Azure

Friday, May 19, 2023 4:43 PM

Link: [Tutorial: Configure dynamic inventories of your Azure resources using Ansible]()

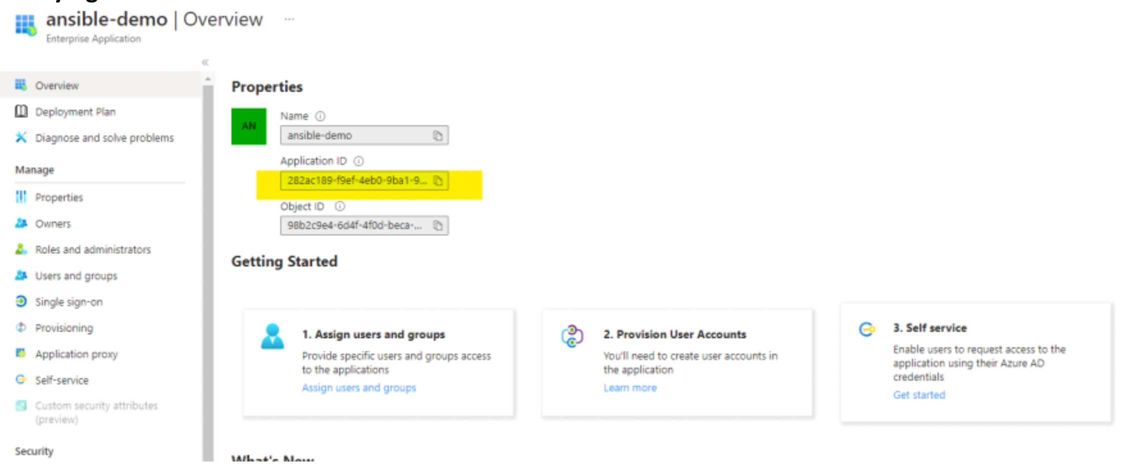
Link: [https://learn.microsoft.com/en-us/azure/developer/ansible/create-ansible-service-principal?tabs=azure-cli]()

1. **Create an Azure service principal:**
   Command: az ad sp create-for-rbac --name ansible --role Contributor --scopes /subscriptions/6923f8cc-4638-4832-a0e7-63be3ca5c15b



```
PS /home/jitendra> az ad sp create-for-rbac --name ansible-demo --role Contributor --scopes /subscriptions/6923f8cc-4638-4832-a0e7-63be3ca5c15b
Creating 'Contributor' role assignment under scope '/subscriptions/6923f8cc-4638-4832-a0e7-63be3ca5c15b'
The output includes credentials that you must protect. Be sure that you do not include these credentials in your code or check the credentials into
{
  "appId": "282ac189-f9ef-4eb0-9ba1-994547e5204f",
  "displayName": "ansible-demo",
  "password": "G0b8Q~We4y1AlRac9w5A~6hONvnnq~UJB~qB5c-7",
  "tenant": "5659fac0-8e34-40af-86b2-dfcd9b0ddbf3"
}
PS /home/jitendra> 
```

2. **Verifying:**


ansible-demo | Overview
Enterprise Application
Overview
Deployment Plan
Diagnose and solve problems
Manage
Properties
Owners
Roles and administrators
Users and groups
Single sign-on
Provisioning
Application proxy
Self-service
Custom security attributes (preview)
Security
AN
Getting Started
1. Assign users and groups
Provide specific users and groups access to the applications
Assign users and groups
2. Provision User Accounts
You'll need to create user accounts in the application
Learn more
3. Self service
Enable users to request access to the application using their Azure AD credentials
Get started
What's New

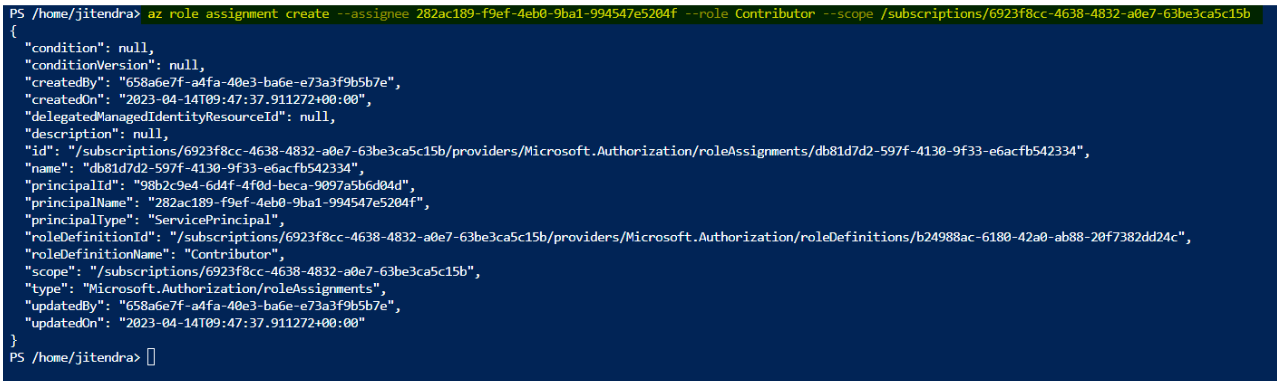
Properties

Name: ansible-demo

Application ID: 282ac189-f9ef-4eb0-9ba1-9...

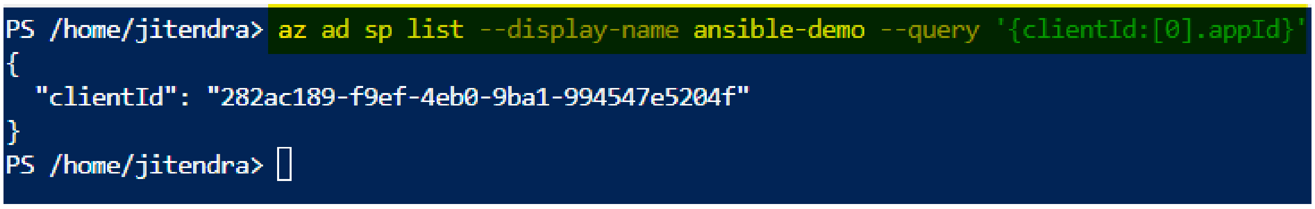
Object ID: 98b2c9e4-6d4f-4f0d-beca-...

3. **Assign a role to the Azure service principal**:
   Command: az role assignment create --assignee 282ac189-f9ef-4eb0-9ba1-994547e5204f --role Contributor --scope /subscriptions/6923f8cc-4638-4832-a0e7-63be3ca5c15b



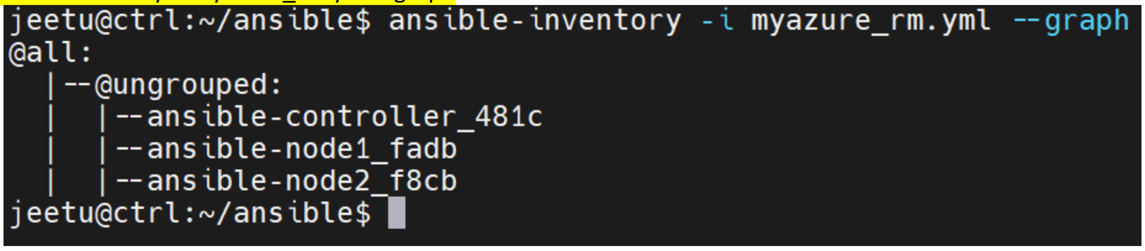
```
PS /home/jitendra> az role assignment create --assignee 282ac189-f9ef-4eb0-9ba1-994547e5204f --role Contributor --scope /subscriptions/6923f8cc-4638-4832-a0e7-63be3ca5c15b
{
  "condition": null,
  "conditionVersion": null,
  "createdBy": "658a6e7f-a4fa-40e3-ba6e-e73a3f9b5b7e",
  "createdOn": "2023-04-14T09:47:37.911272+00:00",
  "delegatedManagedIdentityResourceId": null,
  "description": null,
  "id": "/subscriptions/6923f8cc-4638-4832-a0e7-63be3ca5c15b/providers/Microsoft.Authorization/roleAssignments/db81d7d2-597f-4130-9f33-e6acfb542334",
  "name": "db81d7d2-597f-4130-9f33-e6acfb542334",
  "principalId": "98b2c9e4-6d4f-4f0d-beca-9097a5b6d04d",
  "principalName": "282ac189-f9ef-4eb0-9ba1-994547e5204f",
  "principalType": "ServicePrincipal",
  "roleDefinitionId": "/subscriptions/6923f8cc-4638-4832-a0e7-63be3ca5c15b/providers/Microsoft.Authorization/roleDefinitions/b24988ac-6180-42a0-ab88-20f7382dd24c",
  "roleDefinitionName": "Contributor",
  "scope": "/subscriptions/6923f8cc-4638-4832-a0e7-63be3ca5c15b",
  "type": "Microsoft.Authorization/roleAssignments",
  "updatedBy": "658a6e7f-a4fa-40e3-ba6e-e73a3f9b5b7e",
  "updatedOn": "2023-04-14T09:47:37.911272+00:00"
}
PS /home/jitendra> 
```

4. **Verifying the SP:**
   Command: az ad sp list --display-name ansible-demo --query '{clientId:[0].appId}'



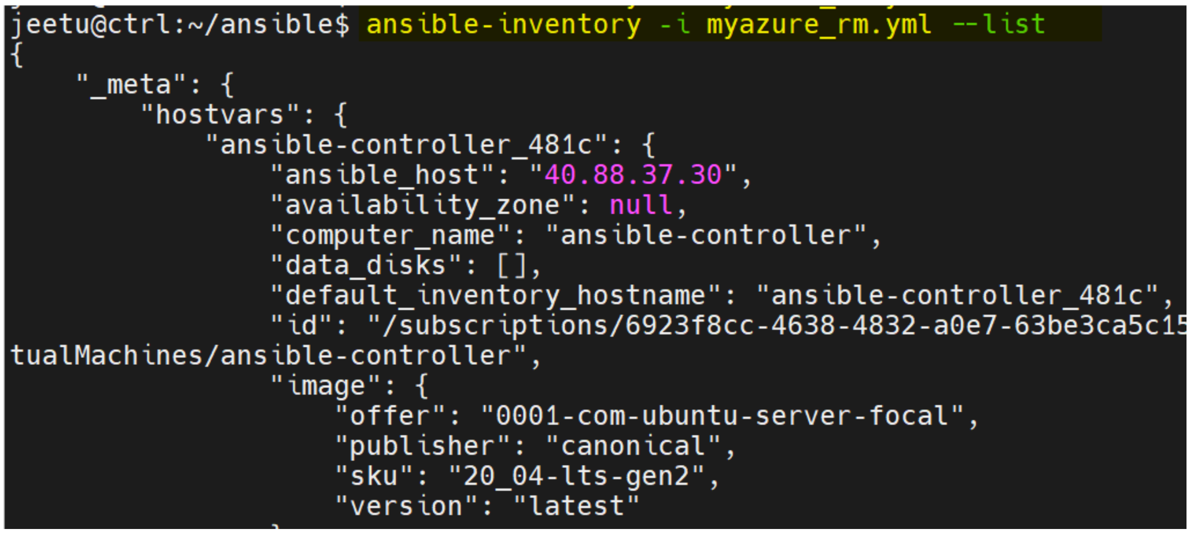
```
PS /home/jitendra> az ad sp list --display-name ansible-demo --query '{clientId:[0].appId}'
{
  "clientId": "282ac189-f9ef-4eb0-9ba1-994547e5204f"
}
PS /home/jitendra> 
```

5. **Install AZ-CLI on Ubuntu:**
   # sudo apt install azure-cli -y
   # az login
   # ansible-inventory -i myazure_rm.yml --graph



```
jeetu@ctrl:~/ansible$ ansible-inventory -i myazure_rm.yml --graph
@all:
  |--@ungrouped:
  |  |--ansible-controller_481c
  |  |--ansible-node1_fadb
  |  |--ansible-node2_f8cb
jeetu@ctrl:~/ansible$ 
```

6. **Fetching the VMs detailed info:**
   Command: ansible-inventory -i myazure_rm.yml --list

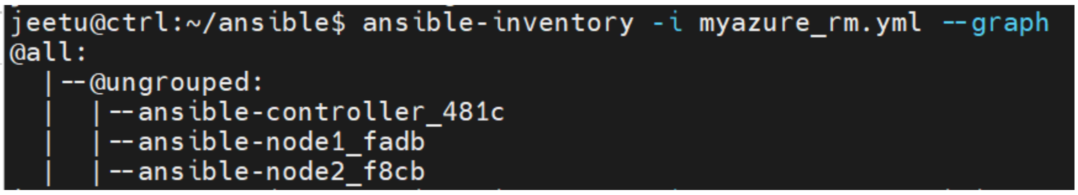


```
jeetu@ctrl:~/ansible$ ansible-inventory -i myazure_rm.yml --list
{
    "_meta": {
        "hostvars": {
            "ansible-controller_481c": {
                "ansible_host": "40.88.37.30",
                "availability_zone": null,
                "computer_name": "ansible-controller",
                "data_disks": [],
                "default_inventory_hostname": "ansible-controller_481c",
                "id": "/subscriptions/6923f8cc-4638-4832-a0e7-63be3ca5c15
tualMachines/ansible-controller",
                "image": {
                    "offer": "0001-com-ubuntu-server-focal",
                    "publisher": "canonical",
                    "sku": "20_04-lts-gen2",
                    "version": "latest"
```

7. **Assign group membership with conditional_groups**
   Open the myazure_rm.yml dynamic inventory and add the following conditional_group:

```
#vim myazure_rm.yml
    plugin: azure_rm
    include_vm_resource_groups:
      - ansible-inventory-test-rg
    auth_source: auto
    conditional_groups:
      linux: "'CentOS' in image.offer"
      windows: "'WindowsServer' in image.offer"
:wq!
```

   Command: ansible-inventory -i myazure_rm.yml --graph



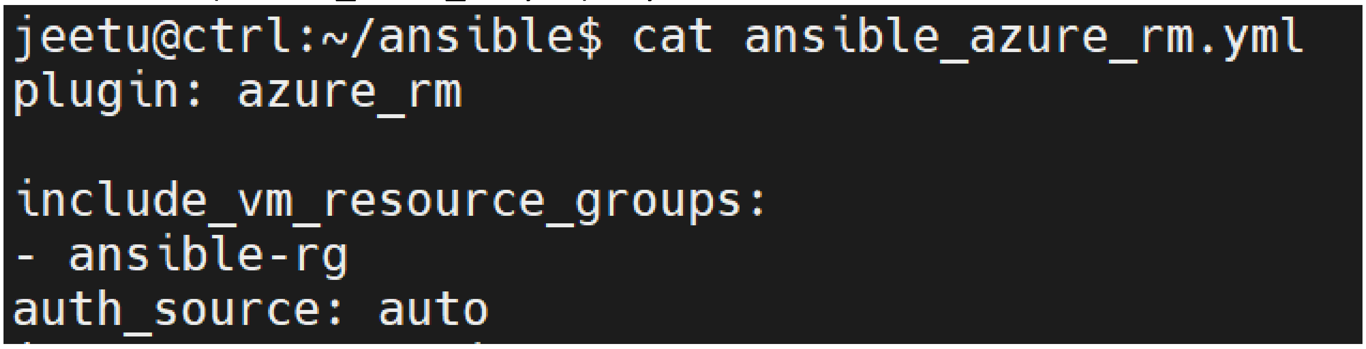
```
jeetu@ctrl:~/ansible$ ansible-inventory -i myazure_rm.yml --graph
@all:
  |--@ungrouped:
  |  |--ansible-controller_481c
  |  |--ansible-node1_fadb
  |  |--ansible-node2_f8cb
```

15. Dynamically fetching inventory from Azure RG:

Friday, May 19, 2023 4:44 PM

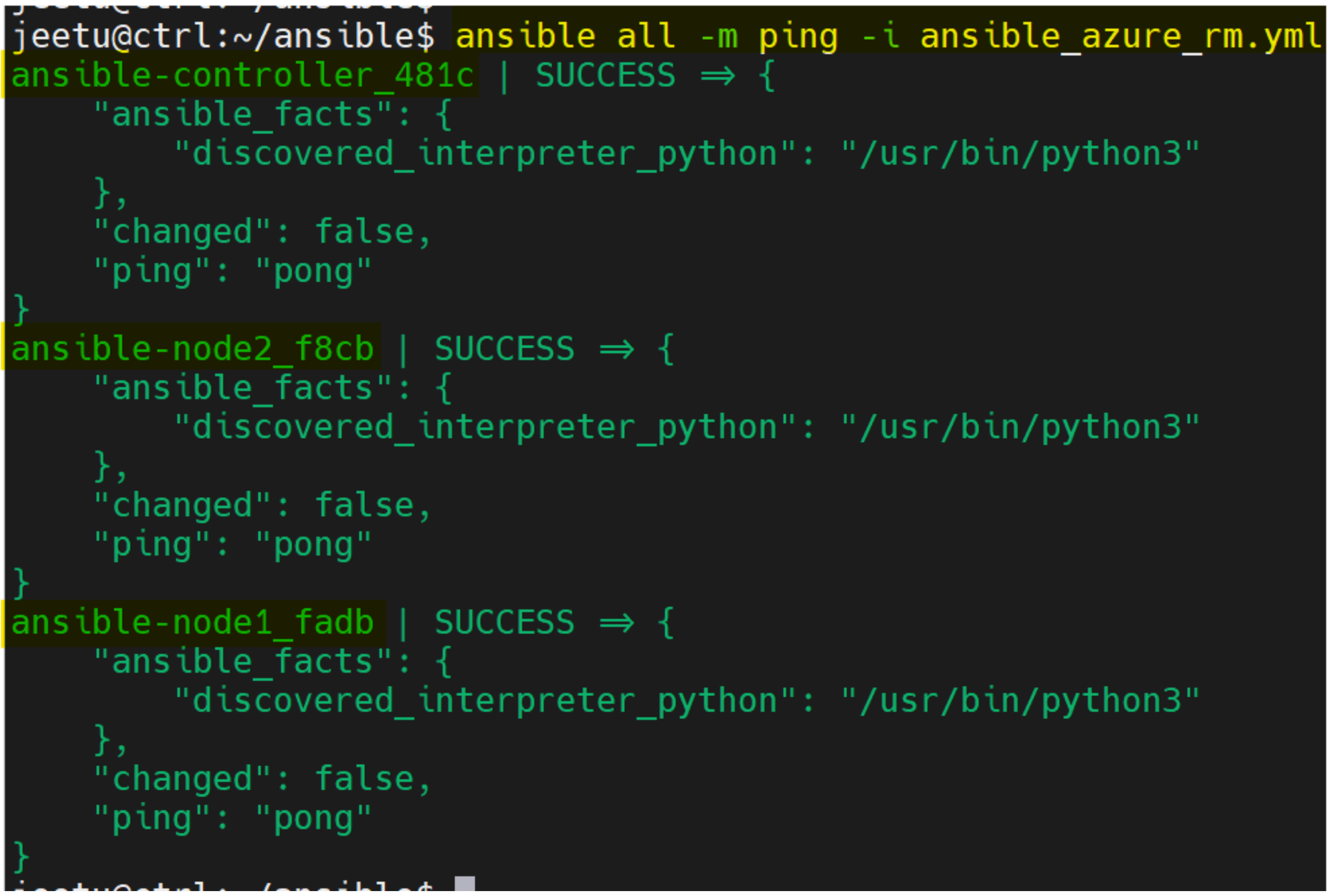
1. Login to Azure portal:
   # az login

2. Create a file (ansible_azure_rm.yml) anywhere:



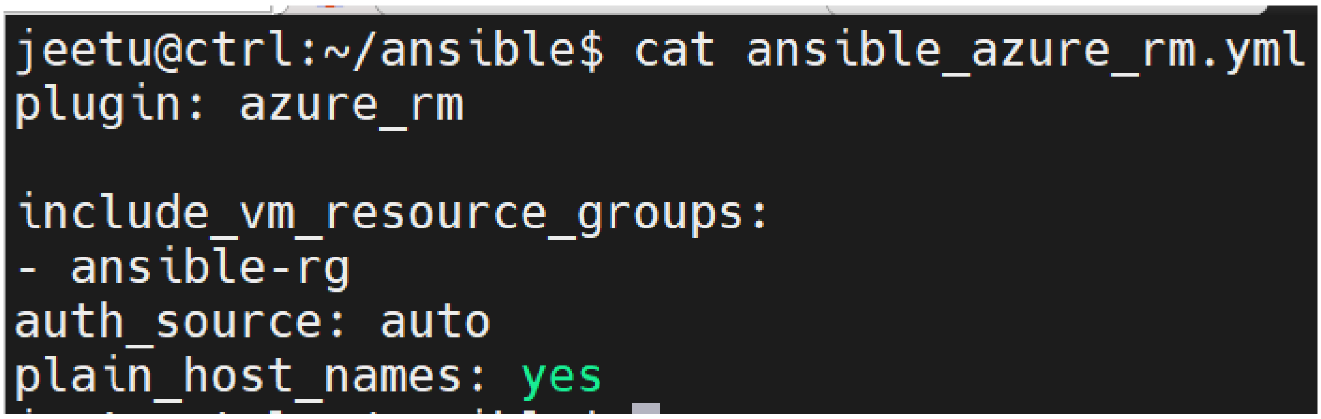
```
jeetu@ctrl:~/ansible$ cat ansible_azure_rm.yml
plugin: azure_rm

include_vm_resource_groups:
- ansible-rg
auth_source: auto
```

Output:



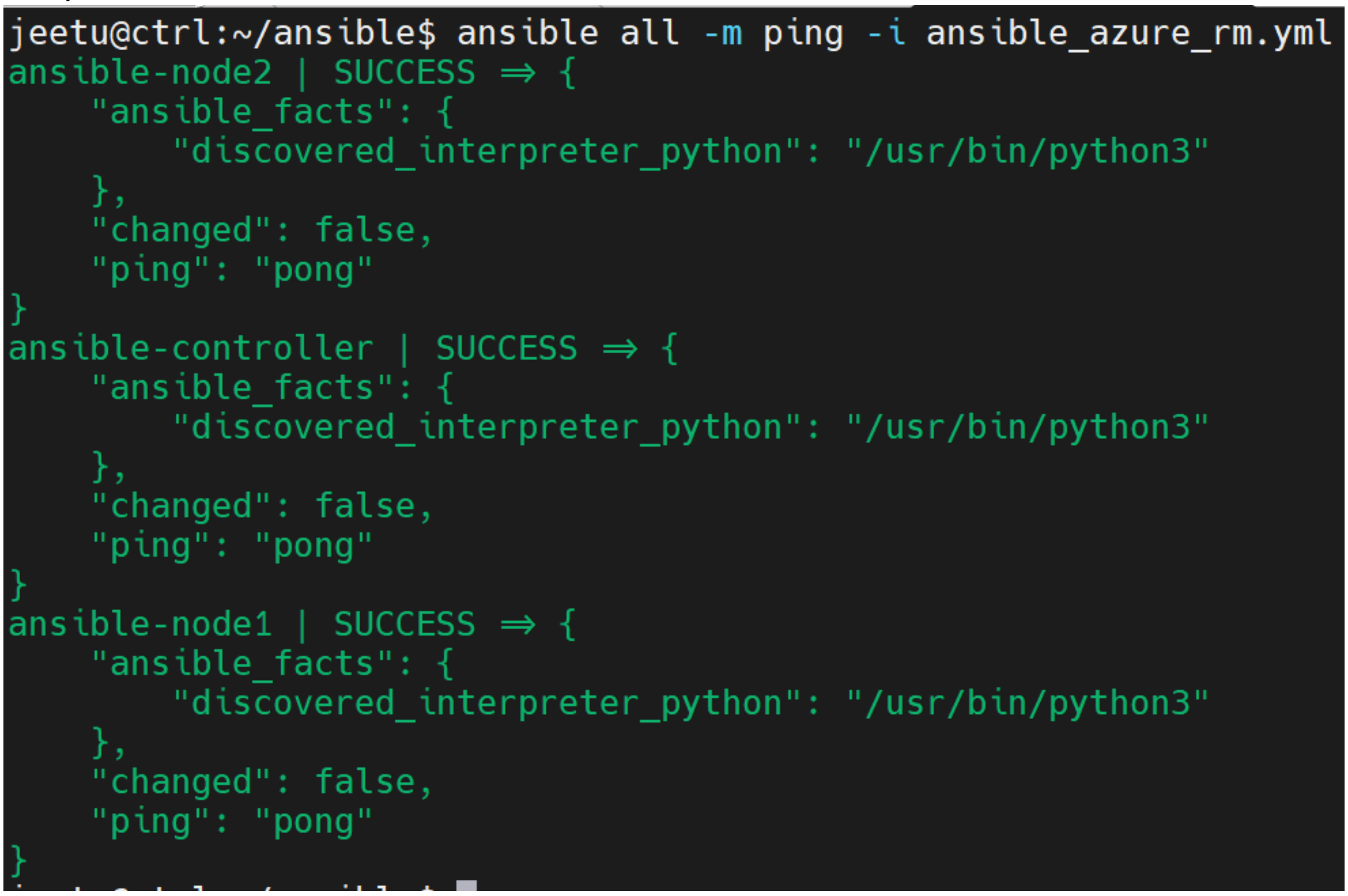
```
jeetu@ctrl:~/ansible$ ansible all -m ping -i ansible_azure_rm.yml
ansible-controller_481c | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
ansible-node2_f8cb | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
ansible-node1_fadb | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
```

But, this is listing NIC card name, instead of VM name. To list actual VM names:



```
jeetu@ctrl:~/ansible$ cat ansible_azure_rm.yml
plugin: azure_rm

include_vm_resource_groups:
- ansible-rg
auth_source: auto
plain_host_names: yes
```

Output:



```
jeetu@ctrl:~/ansible$ ansible all -m ping -i ansible_azure_rm.yml
ansible-node2 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
ansible-controller | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
ansible-node1 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
```

Reference link: https://www.shudnow.io/2019/12/12/ansible-dynamic-inventories-in-azure-part-1/