Ansible Overview

- Ansible is an open-source platform by Red Hat that automates cloud provisioning, configuration management, and application deployments.
- Using Ansible, you can provision VMs, containers, and your entire cloud infrastructure.
- Unlike Puppet or Chef, Ansible is **agentless**.

Ansible Playbooks.

- Playbooks are ordered lists of tasks that have been saved so you can run them repeatedly in the same order. Playbooks are Ansible's language for configuration, deployment, and orchestration. Playbooks are coded using YAML so as to be human-readable.

You run a playbook using the following command:

```
ansible-playbook < playbook name >
```

You can also check the syntax of a playbook using the following command.

```
ansible-playbook --syntax-check
```

To see a list of hosts that would be affected by running a playbook, run the command:

```
ansible-playbook playbook.yml --list-hosts
```

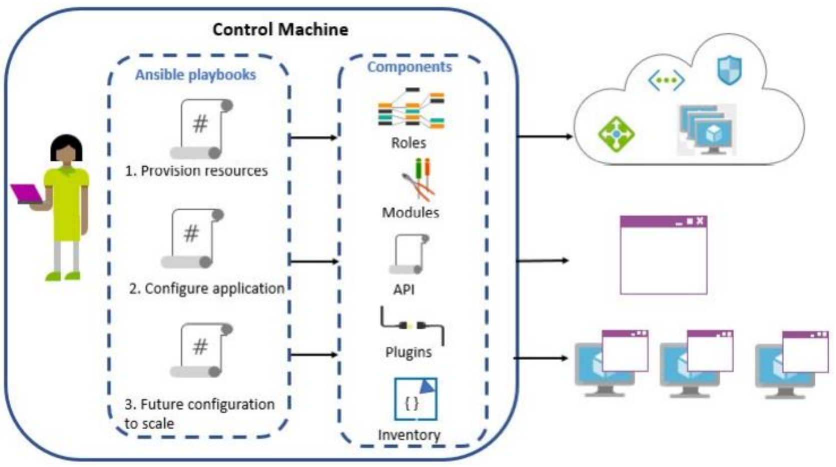
Ansible Modules.

- A playbook is typically made up of many modules. For example, you could have one playbook containing three modules: a module for creating an Azure Resource group, a module for creating a virtual network, and a module for adding a subnet.
- Ansible works by connecting to your nodes, and then pushing small programs (or units of code)—called modules—out to the nodes. Modules are the units of code that define the configuration. They represent the desired state of the system (declarative), are executed over SSH by default, and are removed when finished.
- For interacting with Azure services, Ansible includes a suite of Ansible cloud modules. These modules enable you to create and orchestrate your infrastructure on Azure.

```
- hosts: localhost
  connection: local
  tasks:
    - name: Creating resource group - "{{ name }}"
      azure_rm_resourcegroup:
        name: "{{ name }}"
        location: "{{ location }}"
      register: rg
    - debug:
        var: rg
```

Ansible Workflow:

The following workflow and component diagram outlines how playbooks can run in different circumstances, one after another. In the workflow, Ansible playbooks:



- **Provision resources.** Playbooks can provision resources for example load-balancer, virtual networks, network security groups, and VM scale sets on Azure.
- **Configure the application.** Playbooks can deploy applications to run particular services, such as installing Apache Tomcat on a Linux machine to allow you to run a web application.
- **Future configurations to scale.** Playbooks can alter configurations by applying playbooks to existing resources and applications—in this instance to scale the VMs

In all cases, Ansible makes use of core components such as roles, modules, APIs, plugins and inventory

You don't need to maintain and run commands from any particular central server. Instead, there is a control machine with Ansible installed, and from which playbooks are run.

Installing Ansible

An Ansible installation has the following characteristics:

- You only need to install Ansible on one machine, which could be a workstation or a laptop. You can manage an entire fleet of remote machines from that central point.
- No database is installed as part of the Ansible setup.
- No daemons are required to start or keep Ansible running

Working with Cloud Shell

```
az account list
# Ansible uses by exporting the AZURE_SUBSCRIPTION_ID environment variable.
export AZURE_SUBSCRIPTION_ID=<your-subscription-id>
```

Create and upload file: create_rg.yml

```
- hosts: localhost
```

```
 connection: local
 tasks:
  - name: Creating resource group - "{{ name }}"
    azure_rm_resourcegroup:
     name: "{{ name }}"
     location: "{{ location }}"
    register: rg
  - debug:
     var: rg
```

Run the playbook

```
ansible-playbook create_rg.yaml --extra-vars "name=demo_rg location=eastus"
```

Note: Due to the **register** variable and **debug** section of the playbook, the results display when the command finishes.

Delete the Azure Resource Group

```
- hosts: localhost
 tasks:
  - name: Deleting resource group - "{{ name }}"
    azure_rm_resourcegroup:
     name: "{{ name }}"
     state: absent
    register: rg
  - debug:
     var: rg
```

✓ Note: The Windows operating system is not supported as a control machine. However, you can run Ansible from a Windows machine by utilizing other services and products such as Windows Subsystem for Linux, Azure Cloud Shell, and Visual Studio Code

Steps to Configure a VM to Install / Configure for Ansible (Control Machine)

1. Create an Azure Service Principal and note the App ID and App Secret.
2. Create a Ubuntu VM and Configure Ansible.
    a. Create a Linux Ubuntu VM and Install Ansible on it
        i. SSH to VM: SSH dssadmin@20.185.191.3

           If the VM is Ubuntu Linux

```
# Update all packages that have available updates.
```

```
sudo apt-get update && sudo apt dist-upgrade -y
# Install Python 3 and pip
sudo apt-get install -y libssl-dev libffi-dev python-dev python3-pip
```

If the VM is CentOS Linux

```
# Update all packages that have available updates.
sudo yum update -y
# Install Python 3 and pip.
sudo yum install -y python3-pip
```

ii. Install Ansible.

```
sudo pip3 install ansible
```

iii. Install Ansible modules and plugins for interacting with Azure.

```
sudo ansible-galaxy collection install azure.azcollection
```

iv. Install required modules for Ansible on Azure

```
wget https://raw.githubusercontent.com/ansible-collections/azure/dev/requirements-
azure.txt
```

v. # Install Ansible modules

```
sudo pip3 install -r requirements-azure.txt
sudo ansible --version
```

b. Create a directory named .azure in the home directory and a credentials file under it.

```
mkdir ~/.azure
nano ~/.azure/credentials
```

c. Insert the following lines into the **credentials** file. Replace the placeholders with the information from the service principal details you copied in the previous task. Press **Ctrl+O** to save the file and **Ctrl+X** to exit from the text editor.

```
[default]
subscription_id=f9baec73-91eb-4458-bd0d-965c1973526d
client_id=c1c5e79f-364b-4194-a49b-af9eec02a1dd
secret=f8lV-~mH1t6eqCnkzLnR1-IpsIc_z-v0~7
tenant=82d8af3b-d3f9-465c-b724-0fb186cc28c7
```

Ansible is an agentless architecture based automation tool . Only it needs ssh authentication using Ansible Control Machine private/public key pair. Now let us create a pair of private and public keys. Run the following command to generate a private/public key pair for ssh and to install the public key in the local machine.

```
ssh-keygen -t rsa
chmod 755 ~/.ssh (Allow everyone read and execute – owner can also write)
touch ~/.ssh/authorized_keys (Creates a File for keys)
chmod 644 ~/.ssh/authorized_keys (only owner can read and write and others can read only)
```

```
ssh-copy-id dssadmin@127.0.0.1
yes
<enter password>
```

Note: https://chmod-calculator.com/

d. In the next task, you need SSH private key to created SSH endpoint in Azure DevOps service. Run the following command to get the private key. Copy the private key to notepad.

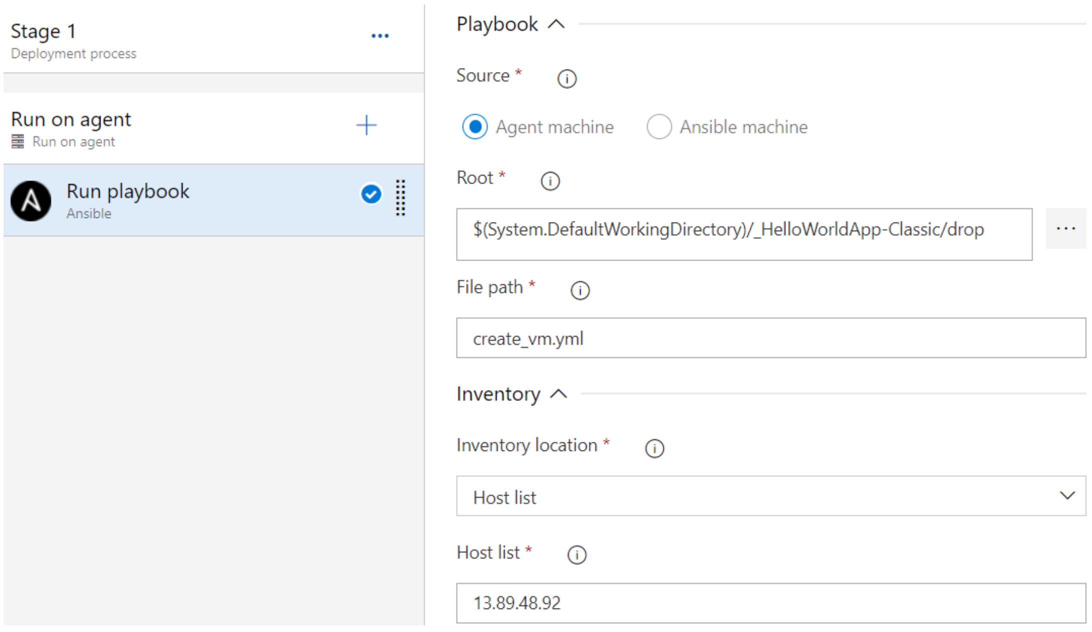
```
cat ~/.ssh/id_rsa
```

3. Create a SSH Service Connection in Azure DevOps
   a. Project Properties → Service Connection → New Service Connection → SSH
   b. In **Add an SSH service connection** window provide the required details and click **OK** to save the connection.
   c. Paste the private key which you copied in the previous task.
4. Add the Ansible Playbook create_vm.yml File to the repository

```
- name: Create Azure VM
 hosts: localhost
 connection: local
 vars:
  resource_group: ansible_rg
  location: eastus
 tasks:
 - name: Create resource group
  azure_rm_resourcegroup:
   name: "{{ resource_group }}"
   location: "{{ location }}"

 - name: Create virtual network
  azure_rm_virtualnetwork:
   resource_group: "{{ resource_group }}"
   name: myVnet
   address_prefixes: "10.0.0.0/16"

 - name: Add subnet
  azure_rm_subnet:
   resource_group: "{{ resource_group }}"
   name: mySubnet
   address_prefix: "10.0.1.0/24"
   virtual_network: myVnet
```

```yaml
- name: Create public IP address
  azure_rm_publicipaddress:
    resource_group: "{{ resource_group }}"
    allocation_method: Static
    name: myPublicIP
  register: output_ip_address

- name: Dump public IP for VM which will be created
  debug:
    msg: "The public IP is {{ output_ip_address.state.ip_address }}."

- name: Create Network Security Group that allows SSH
  azure_rm_securitygroup:
    resource_group: "{{ resource_group }}"
    name: myNetworkSecurityGroup
    rules:
      - name: SSH
        protocol: Tcp
        destination_port_range: 22
        access: Allow
        priority: 1001
        direction: Inbound

- name: Create virtual network inteface card
  azure_rm_networkinterface:
    resource_group: "{{ resource_group }}"
    name: myNIC
    virtual_network: myVnet
    subnet_name: mySubnet
    security_group: myNetworkSecurityGroup
    ip_configurations:
      - name: ipconfig1
        public_ip_address_name: myPublicIP
        primary: True


- name: Create VM
  azure_rm_virtualmachine:
    resource_group: "{{ resource_group }}"
    name: CreatedbyAnsible-vm
    vm_size: Standard_DS1_v2
    admin_username: azureuser
    ssh_password_enabled: true
    admin_password: Password@123
    network_interfaces: myNIC
    image:
      offer: CentOS
      publisher: OpenLogic
      sku: '7.5'
      version: latest
```

5. In the build pipeline add a step to copy the YML to **$(build.outputstagingdirectory)**
6. Run the build pipeline
7. Create / Edit a Release Pipeline and Add the following steps
    a. **Replace Tokens** to replace any tokens if used in Playbook file
    b. **Ansible Task**: This task is to integrate with Ansible. This task executes a given Ansible playbook on a specified list of inventory nodes via command line interface. This task requires that the Playbook files be located either on a private Linux agent or on a remote machine where Ansible automation engine has been installed. Select Ansible Location as **Remote Machine** and select **Ansible SSH endpoint** that you created in **Task 3**.
    c. Under the **Inventory** section, select **Host list** as inventory location and enter **pubic ip** of your ansible vm in **Host list** field.


Stage 1
Deployment process
Run on agent
Run on agent
Run playbook
Ansible

Playbook
Source *
(●) Agent machine ( ) Ansible machine
Root *
$(System.DefaultWorkingDirectory)/_HelloWorldApp-Classic/drop
File path *
create_vm.yml
Inventory
Inventory location *
Host list
Host list *
13.89.48.92

Ansible to **deallocate (stop)** an Azure virtual machine.

```
- name: Deallocate Azure VM
 hosts: localhost
 connection: local
 tasks:
  - name: Stop virtual machine
   azure_rm_virtualmachine:
    resource_group: {{ resource_group_name }}
    name: {{ vm_name }}
    allocated: no
```

Note: In the same script you can delete **allocated property** to start the VM.

Ansible Script to Login, Pull and Run Docker Image

```
- name: Log into DockerHub
 docker_login:
  username: sandeepsoni
  password: XXXXXXXX

- name: pull an image
 docker_image:
  name: sandeepsoni/helloworldapp
  source: pull

- name: Create a data container
 docker_container:
  name: mydata
  image: sandeepsoni/helloworldapp
  volumes:
   - /data
```