Mohammad Saeed Pourmazar

https://github.com/MohammadSaeedPourmazar

https://gitlab.com/MohammadSaeedPourmazar

https://medium.com/@MohammadSaeedPourmazar

https://dev.to/MohammadSaeedPourmazar

https://www.youtube.com/@MohammadSaeedPourmazar

https://www.instagram.com/MohammadSaeedPourmazar

https://www.facebook.com/MohammadSaeedPourmazar

https://www.linkedin.com/in/MohammadSaeedPourmazar/

https://orcid.org/0009-0008-9383-419X

Ansible Playbook (Installing Kubernetes)

Step 1: Setup Your Inventory File:

sudo nano inventory.ini

Modify this file to include the IPs of your Kubernetes nodes.

You need to modify your inventory file to properly define the Kubernetes master and worker nodes. Since you have an Ansible control node (ansible_vm) and a set of Ubuntu VMs (ubuntu_vms), you should classify them correctly for your Kubernetes setup.

[ansible_vm]

IP Address(Ansible) ansible_user=ubuntu ansible_ssh_private_key_file=~/.ssh/id_rsa

[k8s_master]

IP Address(Master)

[k8s_workers]

IP Address(Worker)

[kubernetes:children]

k8s_master

k8s_workers

[kubernetes:vars]

ansible_user=user

ansible_ssh_private_key_file=~/.ssh/id_rsa

ansible_python_interpreter=/usr/bin/python3

[web]

IP Address(Master)

IP Address(Worker)

Changes & Explanation:

1. Removed [ubuntu_vms]:

Your original [ubuntu_vms] group is no longer needed because we are now using [k8s_master] and [k8s_workers].

2. *Created [k8s_master]:*

Add your IP Address(Master) here.

2. Created [k8s_workers] group:

If you have IP Address(worker) , add them here.

3. Created [Kubernetes:vars] group:

A parent group that includes both master and workers.

2. Removed [ubuntu_vms:vars]:

because its settings were already in [kubernetes:vars]:

4. Kept [web] group (if your Kubernetes cluster also runs a web service).

Step 2: Create the Ansible Playbook (install_k8s.yml):

First Create the Ansible Playbook

sudo nano install_k8s.yml

Then modify the installation file (The Exact Version v1.30.0-1.1):

```
---
- name: Install Kubernetes v1.30.0-1.1
  hosts: kubernetes
  become: yes
  tasks:
    - name: Update and upgrade system packages
      apt:
        update_cache: yes
        upgrade: yes

    - name: Set Hostname for Nodes
      hostname:
        name: "{{ 'Master.Node' if inventory_hostname in groups['k8s_master'] else 'Worker.Node' }}"

    - name: Update /etc/hosts file
      blockinfile:
        path: /etc/hosts
        block: |
          {{ ansible_host }} {{ 'Master.Node' if inventory_hostname in groups['k8s_master'] else
'Worker.Node' }}
        marker: "# {mark} ANSIBLE MANAGED BLOCK - KUBERNETES NODES"

    - name: Disable Swap
      command: swapoff -a
      changed_when: false
```

```
- name: Remove swap entry from /etc/fstab
  replace:
    path: /etc/fstab
    regexp: '^(.*swap.*)$'
    replace: '# \1'

- name: Load required kernel modules
  copy:
    dest: /etc/modules-load.d/containerd.conf
    content: |
      overlay
      br_netfilter

- name: Apply kernel modules
  modprobe:
    name: "{{ item }}"
    state: present
  loop:
    - overlay
    - br_netfilter

- name: Set Kernel Parameters for Kubernetes
  copy:
    dest: /etc/sysctl.d/kubernetes.conf
    content: |
      net.bridge.bridge-nf-call-ip6tables = 1
      net.bridge.bridge-nf-call-iptables = 1
      net.ipv4.ip_forward = 1

- name: Apply sysctl parameters
  command: sysctl --system
```

```
- name: Install dependencies for Docker
  apt:
    name:
      - apt-transport-https
      - ca-certificates
      - curl
      - software-properties-common
    state: present

- name: Add Docker GPG key
  ansible.builtin.get_url:
    url: https://download.docker.com/linux/ubuntu/gpg
    dest: /etc/apt/trusted.gpg.d/docker.asc

- name: Add Docker repository
  apt_repository:
    repo: "deb [arch=amd64] https://download.docker.com/linux/ubuntu {{
ansible_distribution_release }} stable"
    state: present

- name: Update package list
  apt:
    update_cache: yes

- name: Install Docker CE
  apt:
    name: docker-ce
    state: present

- name: Enable and start Docker service
  systemd:
    name: docker
```

```
    state: started
    enabled: yes

- name: Install containerd runtime
  apt:
    name: containerd.io
    state: present

- name: Configure containerd
  shell: |
    containerd config default | tee /etc/containerd/config.toml >/dev/null
    sed -i 's/SystemdCgroup = false/SystemdCgroup = true/g' /etc/containerd/config.toml
    systemctl restart containerd
    systemctl enable containerd
  args:
    executable: /bin/bash

- name: Install Kubernetes
  apt:
    name:
      - kubelet=1.30.0-1.1
      - kubeadm=1.30.0-1.1
      - kubectl=1.30.0-1.1
    state: present

- name: Hold Kubernetes packages to prevent automatic upgrades
  command: apt-mark hold kubelet kubeadm kubectl

- name: Ensure Kubernetes services are enabled
  systemd:
    name: "{{ item }}"
    enabled: yes
```

```
  loop:
    - kubelet

- name: Reset Kubernetes if previously installed (Master Node Only)
  command: kubeadm reset -f
  when: inventory_hostname in groups['k8s_master']
  ignore_errors: yes
  register: kubeadm_reset
  failed_when: kubeadm_reset.rc != 0 and "reset" not in kubeadm_reset.stdout

- name: Remove Kubernetes directories (Master Node Only)
  file:
    path: "{{ item }}"
    state: absent
  loop:
    - /etc/kubernetes
    - /var/lib/etcd
  when: inventory_hostname in groups['k8s_master']

- name: Initialize Kubernetes Cluster (Master Node Only)
  command: kubeadm init --pod-network-cidr=10.10.0.0/16
  when: inventory_hostname in groups['k8s_master']
  register: kubeadm_init
  ignore_errors: yes
  failed_when: kubeadm_init.rc != 0 and "initialized" not in kubeadm_init.stdout

- name: Check if admin.conf exists
  stat:
    path: /etc/kubernetes/admin.conf
  register: admin_conf_stat
  when: inventory_hostname in groups['k8s_master']
```

```
- name: Ensure .kube directory exists
  become: yes
  become_user: "{{ ansible_user }}"
  file:
    path: "~/.kube"
    state: directory
    mode: 0755
  when: inventory_hostname in groups['k8s_master']

- name: Copy admin.conf to user config
  copy:
    src: /etc/kubernetes/admin.conf
    dest: "~/.kube/config"
    remote_src: yes
    owner: "{{ ansible_user }}"
    group: "{{ ansible_user }}"
    mode: "0600"
  when:
    - inventory_hostname in groups['k8s_master']
    - admin_conf_stat.stat.exists

- name: Verify kubeconfig file
  command: kubectl --kubeconfig ~/.kube/config get nodes
  when: inventory_hostname in groups['k8s_master']
  register: kubeconfig_verify
  ignore_errors: yes

- name: Fail if kubeconfig verification fails
  fail:
    msg: "Failed to verify kubeconfig file. Check the API server and certificates."
  when: kubeconfig_verify.rc != 0
```

```
- name: Verify API server certificate
  command: openssl verify -CAfile /etc/kubernetes/pki/ca.crt /etc/kubernetes/pki/apiserver.crt
  when: inventory_hostname in groups['k8s_master']
  register: api_cert_verify
  ignore_errors: yes

- name: Fail if API server certificate is invalid
  fail:
    msg: "API server certificate is invalid. Check the certificates."
  when: api_cert_verify.rc != 0

- name: Install Calico Network Plugin (Master Node Only)
  shell: |
    curl -O https://raw.githubusercontent.com/projectcalico/calico/v3.25.0/manifests/calico.yaml
    kubectl apply -f calico.yaml
  args:
    chdir: /root
  when: inventory_hostname in groups['k8s_master']

- name: Install NTP to synchronize time
  apt:
    name: ntp
    state: present

- name: Restart and enable NTP service
  systemd:
    name: ntp
    state: restarted
    enabled: yes
```

Step 3: Run the Ansible Playbook:

Run the following command from your Ansible control node:

ansible-playbook -i inventory.ini install_k8s.yml

This will:

Install Kubernetes on all nodes

** Initialize the master node **

** Deploy the Calico network plugin **

** Retrieve the join command **

** Automatically join worker nodes to the cluster **