Configuring a Docker Container with Ansible to Mount a Volume

In-Depth Steps to Configure a Docker Container with Ansible to Mount a Volume

Step 1: **Install Ansible on the Control Node**

- Ensure Ansible is installed on the machine running the playbook (control node).
- For Ubuntu/Debian, run:

```
sudo apt update
sudo apt install ansible -y
```

- For Red Hat/CentOS, use:

```
sudo yum install epel-release -y
sudo yum install ansible -y
```

- Verify installation:

```
ansible --version
```

Step 2: **Install the Docker Collection for Ansible**

- Install the `community.docker` collection to manage Docker resources.
- Run:

```
ansible-galaxy collection install community.docker
```

- Ensure Python's Docker SDK is installed on the control node:

```
pip install docker
```

Step 3: **Set Up the Target Host**

- Ensure the target host (where the Docker container will run) has Docker installed.

- For Ubuntu/Debian, install Docker:

```
sudo apt update
sudo apt install docker.io -y
sudo systemctl enable docker
sudo systemctl start docker
```

- Add the Ansible user to the `docker` group to avoid permission issues:

```
sudo usermod -aG docker <ansible_user>
```

- Replace <ansible$_{u}ser > withtheSSHuser(e.g.,$ ubuntu$).VerifyDocker :$

-

```
docker --version
```

Configure SSH Access

- Ensure the control node can SSH into the target host.
- Generate an SSH key pair on the control node (if not already present):

```
ssh-keygen -t rsa -b 4096
```

- Copy the public key to the target host:

```
ssh-copy-id <ansible_user>@<target_host_ip>
```

- Test SSH connectivity:

```
ssh <ansible_user>@<target_host_ip>
```

Create the Ansible Inventory

- Create a file named `inventory.yml` in the project directory.
- Add the target host details:

```
docker_hosts:
  hosts:
    target_host:
      ansible_host: <target_host_ip>
      ansible_user: <ansible_user>
      ansible_ssh_private_key_file: <path_to_ssh_key>
```

- Replace <target$_{h}ost_{i}p >,$ <ansible$_{u}ser >, and$ <path$_{t}o_{s}sh_{k}ey > withappropriatevalues(e.g.,$ 192

Create the Ansible Playbook

- Create a file named deploy$_{n}ginx.ymlintheprojectdirectory.Addthefollowingcontenttodeploya$

- ---
  - name: Deploy Nginx container with volume mount
    hosts: docker_hosts
    become: yes
    tasks:
      - name: Ensure Docker is installed
        ansible.builtin.apt:
          name: docker.io
          state: present
          update □□_cache: yes
        when: ansible_os_family == ”Debian”

      - name: Ensure Docker service is running
        ansible.builtin.service:
          name: docker
          state: started
          enabled: yes

      - name: Create directory for Nginx logs on host
        ansible.builtin.file:
          path: /opt/nginx_logs
          state: directory
          mode: ’0755’
          owner: root
          group: root

      - name: Deploy Nginx container with volume mount
        community.docker.docker_container:
          name: nginx_container
          image: nginx:latest
          state: started
          restart_policy: always
          ports:
            - ”80:80”
          volumes:
            - /opt/nginx_logs:/var/log/nginx
        register: container_result

      - name: Display container status
        ansible.builtin.debug:
          msg: ”Nginx container deployed with ID: {{ container_result.co

This playbook:

- Targets hosts in the $\texttt{docker}_{h}ostsgroup.InstallsDockeronDebian-basedsystems.$
- Ensures the Docker service is running.

- Creates $\texttt{/opt/nginx}_{l}ogsonthehost.DeploysanNginxcontainer, mapping\texttt{/opt/nginx}_{l}ogsto\texttt{/var/l}$

- **Adjust Permissions for the Volume**
  - Nginx in the container runs as user `nginx` (UID 101).
  - Set permissions on `/opt/nginx`$_{logstoallowthecontainertowritelogs}$:

```
sudo chown 101:101 /opt/nginx_logs
sudo chmod 775 /opt/nginx_logs
```

Add this as an Ansible task if needed, before the container deployment task:

```
- name: Set permissions for Nginx logs directory
  ansible.builtin.file:
    path: /opt/nginx_logs
    state: directory
    mode: '0775'
    owner: 101
    group: 101
```

Run the Ansible Playbook

- From the project directory, execute:

```
ansible-playbook -i inventory.yml deploy_nginx.yml
```

- Ensure the control node has access to the inventory and playbook files.
- Monitor the output for errors (e.g., SSH issues, Docker not running).

Verify the Container Deployment

- SSH into the target host and check running containers:

```
docker ps
```

- Confirm the container `nginx`$_{containerisrunning.Inspectthevolumemount}$:
-

```
docker inspect nginx_container | grep -A 5 Mounts
```

Look for the bind mount mapping `/opt/nginx`$_{logsto}$`/var/log/nginx`.

Verify the Volume Mount

- Check for Nginx log files on the host:

```
ls /opt/nginx_logs
```

- Expected files: `access.log`, `error.log`.
- Generate some traffic to test logging:

```
curl http://<target_host_ip>
```

- Verify new log entries:

```
cat /opt/nginx_logs/access.log
```

Test the Nginx Service

- Open a browser and navigate to `http://<target`$_{h}ost_{i}p > .Confirmthe Nginxwelcomepage$
- If inaccessible, check:
  - Firewall settings (`ufw allow 80` or equivalent).
  - Container status (`docker ps`).

– Port mapping (`docker inspect nginx`$_{c}ontainer).$

Troubleshoot Common Issues

- **Docker Not Found**: Ensure Docker is installed and the service is running.
- **Permission Denied**: Verify `/opt/nginx`$_{l}ogspermissionsandDockergroupmembership.$**Ansi**
  $CheckSSHkeys, inventorysettings, andnetworkconnectivity.$
- **Container Fails to Start**: Review Docker logs:

```
docker logs nginx_container
```

Clean Up (Optional)

- To remove the container:

```
docker rm -f nginx_container
```

- To delete the log directory:

```
sudo rm -rf /opt/nginx_logs
```

- Add cleanup tasks to the playbook if needed:

```
- name: Remove Nginx container
  community.docker.docker_container:
    name: nginx_container
    state: absent
- name: Remove Nginx logs directory
  ansible.builtin.file:
    path: /opt/nginx_logs
    state: absent
```

Notes

- This setup uses a bind mount (`/opt/nginx`$_{logs}$).$For a Docker-managed volume, replace the v$

```
volumes:
  - nginx_logs:/var/log/nginx
```

and create the volume with:

```
docker volume create nginx_logs
```

For non-Debian systems, modify the Docker installation task (e.g., use `yum` for CentOS).

Use `ansible-vault` for sensitive data like SSH keys.

Test the playbook in a development environment before production use.