Laboratory 1

Introduction

The aim of these exercises is to familiarize students with infrastructure that is shared by RTB House. Every student is given access to 10 identical virtual machines within RTB House resources. These machines will be useful during the rest of the laboratories and the final project will be required to run on these resources.

1. VPN Connection to RTB House

Firstly, the connection to the RTB House virtual machines has to be established. Follow the instructions here:
http://mimuw.rtbhouse.com/OpenVPN.pdf

You can also log in with your credentials to the VM management service at https://mimjenkins.rtb-lab.pl/.

This service allows you to restore the selected VM's to their initial state - beware, all data will be lost!

2. Infrastructure Management with Ansible

The aim of this exercise is to create an Ansible playbook that will orchestrate the installation of docker engine on students virtual machines. Later you will have to create an Ansible playbook that will connect the machines in a docker swarm cluster. Finally you will have to run a simple http application as docker swarm service.

2.1 Docker runtime installation

1. Log into one of your machines, preferably the first available one:

```
ssh <username>@<username>vm101.rtb-lab.pl
```

2. Install Ansible and sshpass:

```
sudo apt -y install ansible sshpass
```

3. Add all of your machines to ssh known_hosts:

```
for i in `seq -w 01 10`; do sshpass -p <password> ssh
<username>@<username>vm1$i.rtb-lab.pl -o StrictHostKeyChecking=no -C
"/bin/true"; done
```

4. Edit the hosts file for Ansible to define docker nodes group:

```
sudo nano /etc/ansible/hosts
```

```
[docker]
<username>vm1[01:10].rtb-lab.pl
```

5. Check that Ansible works:

```
ansible docker -m ping  --extra-vars "ansible_user=<username>
ansible_password=<password>"
```

6. Create docker.service override file to enable docker external port access:

```
nano docker.service
```

```
[Service]
ExecStart=
ExecStart=/usr/bin/dockerd -H fd:// -H tcp://0.0.0.0:2375
```

7. Create docker installation playbook file:

```
nano docker-playbook.yaml
```

```
---
- name: Docker install
  hosts: docker

  tasks:
     - name: Install the latest version of Docker
       become: true
       become_user: root
       yum:
             name: docker.io
             state: latest

     - name: Create directory
       become: true
       ansible.builtin.file:
              path: /etc/systemd/system/docker.service.d
              state: directory

     - name: Copy docker service file with owner and permissions
       become: true
       register: service_conf
       ansible.builtin.copy:
             src: docker.service
             dest: /etc/systemd/system/docker.service.d/override.conf
             owner: root
             group: root
             mode: '0644'

     - name: Ensure the docker daemon is enabled
       become: true
       become_user: root
       systemd:
             name: docker
             state: started
             enabled: yes
             daemon_reload: yes

     - name: Restart daemon on config change
       become: true
       become_user: root
       systemd:
             name: docker
             state: restarted
       when: service_conf.changed
```

8. Run docker installation playbook file:

```
ansible-playbook --extra-vars "ansible_user=<username>
ansible_password=<password>" docker-playbook.yaml
```

9. Verify that the local and remote docker runtimes are working:

```
sudo docker info
```

```
DOCKER_HOST=”<username>vm102.rtb-lab.pl” docker info
```

2.2 Docker Swarm cluster

1. Add swarm manager and swarm worker groups to host file:

```
sudo nano /etc/ansible/hosts
```

```
[swarm_manager]
<username>vm101.rtb-lab.pl

[swarm_nodes]
<username>vm1[02:10].rtb-lab.pl
```

2. Create swarm playbook file:

```
nano swarm-playbook.yaml
```

```
---
- name: Docker Swarm init
  any_errors_fatal: true
  hosts: swarm_manager
  remote_user: root

  tasks:
     - name: Ensure the docker daemon is running
       become: true
       become_user: root
       command: systemctl is-active docker
       register: docker_status
       failed_when: docker_status.stdout_lines[0] != "active"

     - name: Init Docker Swarm
       become: true
       become_user: root
       command: docker swarm init --advertise-addr {{ ansible_facts.eth0.ipv4.address }}
       register: swarm_init
       failed_when: swarm_init.rc != 0

     - name: Get join token
       become: true
       become_user: root
       command: docker swarm join-token -q worker
       register: join_token
       failed_when: join_token.rc != 0

- name: Swarm nodes register
```

```
  any_errors_fatal: true
  hosts: swarm_nodes
  remote_user: root

  tasks:
     - name: Ensure the docker daemon is running
       become: true
       become_user: root
       command: systemctl is-active docker
       register: docker_status
       failed_when: docker_status.stdout_lines[0] != "active"

     - name: Join swarm
       become: true
       become_user: root
       command: docker swarm join --token {{
hostvars[groups['swarm_manager'][0]]['join_token']['stdout_lines'][0] }} {{
hostvars[groups['swarm_manager'][0]]['ansible_eth0']['ipv4']['address'] }}:2377
       register: join_status
       failed_when: join_status.rc != 0
```

3. Run swarm installation playbook file:

```
ansible-playbook --extra-vars "ansible_user=<username>
ansible_password=<password>" swarm-playbook.yaml
```

4. Check if swarm is active:

```
sudo docker info
```

2.3 Create simple HTTP service

1. Start the service:

```
sudo docker service create --name hello -p 80:80 -d
nginxdemos/hello:plain-text
```

2. Check if the service works:

```
curl <username>vm101.rtb-lab.pl
```

3. Scale the service to 10 replicas

```
sudo docker service scale hello=10
```

4. Verify that the swarm manager uses ingress load balancing on the service (the address of server is changing on every request):

```
for i in `seq 1 10`; do curl <username>vm101.rtb-lab.pl; done
```

5. Remove the service:

```
sudo docker service rm hello
```