



Déploiement d'OpenStack-Ansible

26.05.2022

Zakariaa Oulhafiane

Partenariat BCP x Ecole 1337

1337

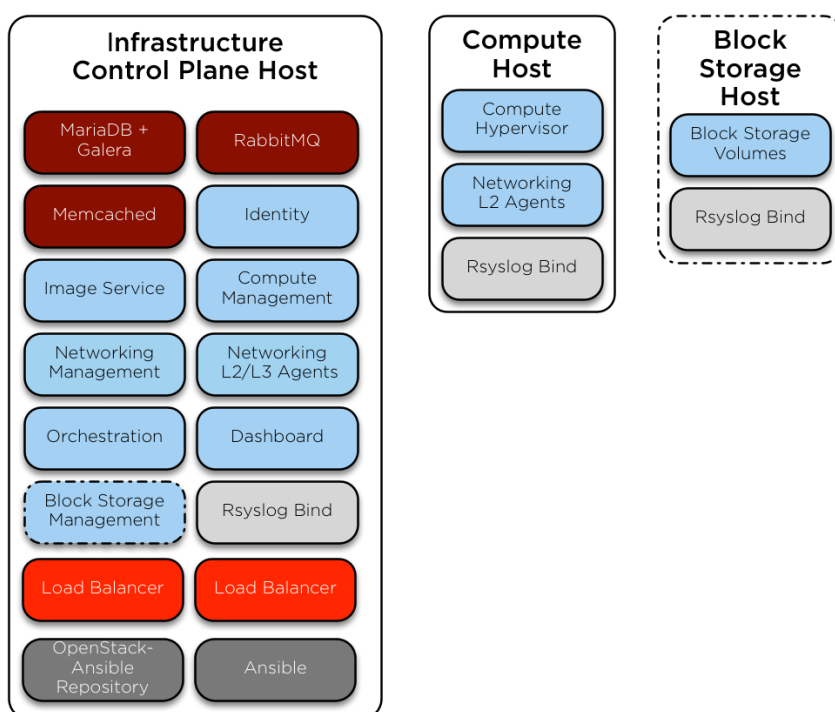


Caractéristiques de l'environnement

Avertissement:

Ceci n'est pas un environnement de production, le but de ce guide est d'acquérir des connaissances sur le déploiement d'OpenStack-Ansible à l'aide d'un environnement de test (architecture à 3 nœuds)

Host and Service Layout - Test Environment



● Infrastructure service
 ● OpenStack service
 ● Logging service
 Optional component

Remarque:

Dans un environnement de production, il est recommandé d'utiliser un hôte de déploiement distinct qui contient Ansible et orchestre l'installation d'OpenStack-Ansible (OSA) sur les hôtes cibles.

L'hôte de déploiement (où Ansible est exécuté) doit être configuré pour être sur le même réseau de couche 2 que le réseau désigné pour la gestion des conteneurs.

Par défaut, il s'agit du réseau br-mgmt.

Architectures réseau:

L'architecture de référence OpenStack-Ansible segmente le trafic à l'aide de VLAN sur plusieurs interfaces ou liaisons réseau.

Réseaux courants utilisés dans un déploiement OpenStack-Ansible peut être observé dans le tableau suivant:

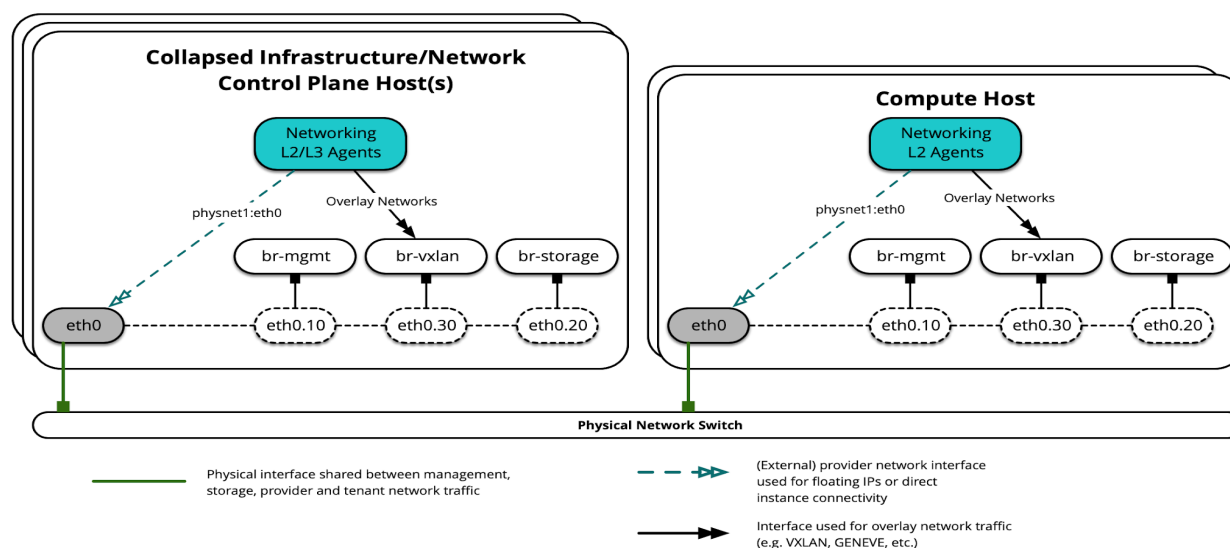
Network	CIDR	Range	VLAN
br-mgmt	172.29.236.0/22	172.29.236.0 - 172.29.239.255	40
br-storage	172.29.244.0/22	172.29.244.0 - 172.29.247.255	20
br-vxlan	172.29.240.0/22	172.29.240.0 - 172.29.243.255	30
br-vlan	10.62.1.0/24	10.62.1.0 - 10.62.1.255	None

- ❖ **br-mgmt:** Le réseau de gestion, également appelé réseau de conteneurs, assure la gestion et la communication entre l'infrastructure et les services OpenStack exécutés dans des conteneurs ou sur du métal.
- ❖ **br-storage:** Le réseau de stockage fournit un accès séparé au stockage par blocs à partir des services OpenStack tels que Cinder et Glance.
- ❖ **br-vxlan:** L'interface br-vxlan est requise si l'environnement est configuré pour permettre aux projets de créer des réseaux virtuels à l'aide de VXLAN. Il fournit l'interface pour le trafic réseau du tunnel virtuel encapsulé (VXLAN).
- ❖ **br-vlan:** Fournit une infrastructure pour les réseaux VLAN étiquetés ou FLAT (sans étiquette VLAN).

Remarque:

br-vlan n'a pas besoin d'une adresse IP car il ne gère que la connectivité de couche 2. Mais dans ce déploiement, nous l'avons utilisé comme interface principale utilisée pour interagir avec le serveur via SSH.

Network Interface Layout - Single Interface



Nos serveurs bare metal:

Dans notre environnement de déploiement de test, nous avons:

Un hôte d'infrastructure (Controller1).

Un hôte de calcul (Compute1).

Un hôte de stockage (Storage1).

Une carte d'interface réseau (NIC) pour chaque hôte.

Les hôtes sont connectés avec d'autres dans la couche 2.

Accès Internet via le réseau: 10.62.1.0/24.

Server Name	IP	br-mgmt-ip	br-vxlan-ip	br-storage-ip	User	Interface
Controller1	10.62.1.1	172.29.236.10	172.29.240.10	172.29.244.10	c1	ens2f1
Compute1	10.62.1.2	172.29.236.11	172.29.240.11	172.29.244.11	compute1	ens2f1
Storage1	10.62.1.3	172.29.236.13	172.29.240.13	172.29.244.13	storage1	eno1

Remarque:

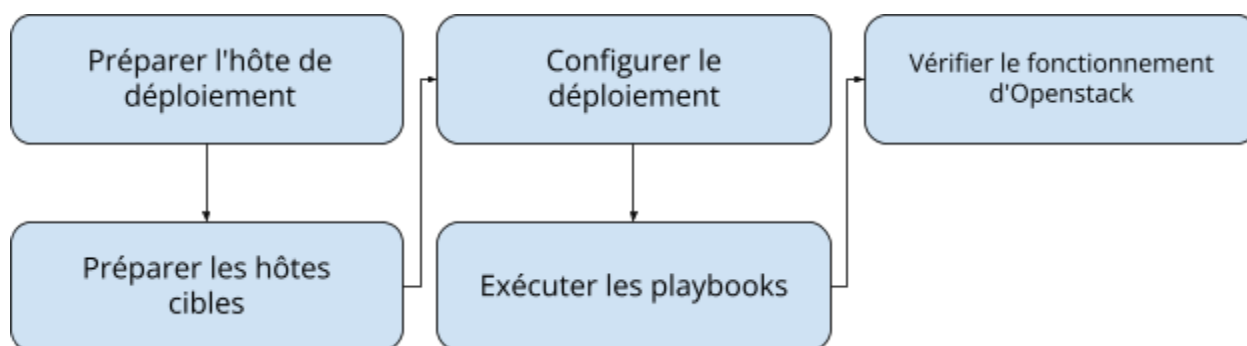
OpenStack-Ansible prend en charge un certain nombre d'architectures réseau différentes et peut être déployé à l'aide **d'une seule interface** réseau pour les charges de travail **hors production** ou à l'aide de plusieurs interfaces réseau ou interfaces liées pour les charges de travail de production.

Catalogue de services:

Nom de Service	Les Services Openstack	Le Role de Services
Nova	Service de calcule	Création de machines virtuelles
		Suppression de machines virtuelles
Glance	Service Image	Télécharger une image
		Mettre à jour une image
		Supprimer une image
Neutron	Service reseau	Création des réseaux
		Création des routeur
		Création des interface réseaux
		Création de l'interconnexion vers un réseau public
Cinder	Service de stockage en bloc	Fournit des volumes
		Fournit des volumes de snapshot
Horizon	Tableau de bord	Gestion de tous les composants OpenStack via une interface web (User-Friendly)

Procédure d'installation

Le diagramme suivant montre le flux de travail général d'une installation OpenStack-Ansible.



Les systèmes d'exploitation pris en charge par OpenStack-Ansible 24.2.0 sont :

- Ubuntu server 22.04 (Focal Fossa) LTS 64-bit (Experimental support in the Yoga release)
- Ubuntu server 20.04 (Focal Fossa) LTS 64-bit
- Debian 10 64-bit
- Centos 8 Stream 64-bit * Derivatives: Rocky Linux

Nous avons choisi Ubuntu 20.04 (Focal Fossa) LTS 64-bit.

Préparer l'hôte de déploiement et les hôtes cibles:

En raison de notre architecture de déploiement 3 nœuds, l'hôte de déploiement est également un hôte cible qui est le nœud de contrôleur (Controller1).

Dans chaque nœud, nous avons installé **ubuntu-server** et mis à jour le noyau au dernier en utilisant les commandes suivantes:

```
# Controller1 & Compute1 & Storage1 #
$ apt-get update
$ apt dist-upgrade
```

Commençons par le nœud **Controller1** (hôte de déploiement et hôte cible):

```
# Controller1 #
$ apt install build-essential git chrony openssh-server python3-dev sudo
$ apt install bridge-utils debootstrap openssh-server tcpdump vlan python3 postfix bsd-mailx
aide-common
$ apt install linux-modules-extra-$(uname -r)
$ vim /etc/netplan/00-installer-config.yaml
```

Ajoutez les configurations suivantes au fichier:

```
network:
  version: 2
  ethernet:
    ens2f1:
      dhcp4: false
  vlans:
    ens2f1.20:
      id: 20
      link: ens2f1
    ens2f1.30:
      id: 30
      link: ens2f1
    ens2f1.40:
      id: 40
      link: ens2f1
  bridges:
    br-mgmt:
      addresses:
        - 172.29.236.10/22
      interfaces:
        - ens2f1.40
      nameservers:
        addresses:
          - [dns-address]
        search:
          - [search-domain]
    br-storage:
      addresses:
        - 172.29.244.10/22
      interfaces:
        - ens2f1.20
    br-vxlan:
      addresses:
        - 172.29.240.10/22
      interfaces:
        - ens2f1.30
    br-vlan:
      addresses:
        - 10.62.1.1/24
      interfaces:
        - ens2f1
      routes:
        - to: default
          via: [gateway-ip]
    br-vlan-veth: {}
  eth12: {}
```

Remarque: Remplacez les [dns-address], [search-domain], et [gateway-ip] par vos adresses, et l'interface ens2f1 par votre interface.

Création de clé privée et clé publique:

```
# Controller1 #
$ ssh-keygen -t rsa -b 4096
$ cp ~/.ssh/id_rsa.pub >> ~/.ssh/authorized_keys
$ cat ~/.ssh/id_rsa.pub
```

Remarque: Copiez le contenu de ~/.ssh/id_rsa.pub et collez-le sur chaque nœud dans ~/.ssh/authorized_keys

```
# Compute1 & Storage1 #
$ mkdir ~/.ssh
$ vim ~/.ssh/authorized_keys
# Copiez le contenu de ~/.ssh/id_rsa.pub du Controller1 #
```

Installation des outils nécessaires sur **Compute1** et **Storage1**

```
# Compute1 & Storage1 #
$ apt install bridge-utils debootstrap openssh-server tcpdump vlan python3
postfix bsd-mailx aide-common
$ apt install linux-modules-extra-$(uname -r)
```

Configuration du stockage dans **Storage1**:

```
# Storage1 #
$ pvcreate --metadatasize 2048 /dev/sdb
$ vgcreate cinder-volumes /dev/sdb
```

Remarque: Remplacez /dev/sdb par le chemin de votre disque physique

Création de la paire Veth sur **tous les hôtes**:

```
# Controller1 & Compute1 & Storage1 #
$ vim /etc/systemd/networkd.conf
```

Ajoutez les configurations suivantes au fichier:

```
[NetDev]
Name=eth12
Kind=veth
[Peer]
Name=br-vlan-veth
```

Vous devez commenter ou supprimer cette entrée 127.0.1.1 hostname dans le fichier /etc/hosts pour éviter les problèmes de résolution de noms.

```
# Controller1 & Compute1 & Storage1 #
$ vim /etc/hosts
#127.0.1.1 hostname
```

Configuration du réseau dans **Compute1**:

```
# Compute1 #  
$ vim /etc/netplan/00-installer-config.yaml
```

Ajoutez les configurations suivantes au fichier:

```
network:  
  version: 2  
  ethernet:  
    ens2f1:  
      dhcp4: false  
  vlans:  
    ens2f1.20:  
      id: 20  
      link: ens2f1  
    ens2f1.30:  
      id: 30  
      link: ens2f1  
    ens2f1.40:  
      id: 40  
      link: ens2f1  
  bridges:  
    br-mgmt:  
      addresses:  
        - 172.29.236.11/22  
      interfaces:  
        - ens2f1.40  
      nameservers:  
        addresses:  
          - [dns-address]  
        search:  
          - [search-domain]  
    br-storage:  
      addresses:  
        - 172.29.244.11/22  
      interfaces:  
        - ens2f1.20  
    br-vxlan:  
      addresses:  
        - 172.29.240.11/22  
      interfaces:  
        - ens2f1.30  
    br-vlan:  
      addresses:  
        - 10.62.1.2/24  
      interfaces:  
        - ens2f1  
      routes:  
        - to: default  
          via: [gateway-ip]  
    br-vlan-veth: {}  
  eth12: {}
```

Remarque: Remplacez les [dns-address], [search-domain], et [gateway-ip] par vos adresses, et l'interface ens2f1 par votre interface.

Configuration du réseau dans **Storage1**:

```
# Storage1 #  
$ vim /etc/netplan/00-installer-config.yaml
```

Ajoutez les configurations suivantes au fichier:

```
network:  
  version: 2  
  ethernet:  
    eno1:  
      dhcp4: false  
  vlans:  
    eno1.20:  
      id: 20  
      link: eno1  
    eno1.30:  
      id: 30  
      link: eno1  
    eno1.40:  
      id: 40  
      link: eno1  
  bridges:  
    br-mgmt:  
      addresses:  
        - 172.29.236.13/22  
      interfaces:  
        - eno1.40  
      nameservers:  
        addresses:  
          - [dns-address]  
        search:  
          - [search-domain]  
    br-storage:  
      addresses:  
        - 172.29.244.13/22  
      interfaces:  
        - eno1.20  
    br-vxlan:  
      addresses:  
        - 172.29.240.13/22  
      interfaces:  
        - eno1.30  
    br-vlan:  
      addresses:  
        - 10.62.1.3/24  
      interfaces:  
        - eno1  
      routes:  
        - to: default  
          via: [gateway-ip]  
    br-vlan-veth: {}  
    eth12: {}
```

Remarque: Remplacez les [dns-address], [search-domain], et [gateway-ip] par vos adresses, et l'interface eno1 par votre interface.

Redémarrez les hôtes et testez la connexion entre eux:

```
# Controller1 & Compute1 & Storage1 #  
$ netplan try  
$ reboot
```

```
# Controller1 #  
$ ping 172.29.236.11  
$ ping 172.29.240.11  
$ ping 172.29.244.11  
$ ping 172.29.236.13  
$ ping 172.29.240.13  
$ ping 172.29.244.13  
$ ping google.com
```

```
# Compute1 #  
$ ping 172.29.236.10  
$ ping 172.29.240.10  
$ ping 172.29.244.10  
$ ping 172.29.236.13  
$ ping 172.29.240.13  
$ ping 172.29.244.13  
$ ping google.com
```

```
# Storage1 #  
$ ping 172.29.236.11  
$ ping 172.29.240.11  
$ ping 172.29.244.11  
$ ping 172.29.236.10  
$ ping 172.29.240.10  
$ ping 172.29.244.10  
$ ping google.com
```

Avertissement:

À ce stade, chaque nœud doit pouvoir se connecter aux autres via les VLANs créés et doit disposer d'une connexion Internet fonctionnelle. Si quelque chose ne fonctionne pas, il doit être corrigé maintenant avant de passer à d'autres étapes.

Installer la source et les dépendances:

```
# Controller1 #
$ mkdir /opt/openstack-ansible
$ git clone -b master https://opendev.org/openstack/openstack-ansible
/opt/openstack-ansible
$ cd /opt/openstack-ansible
$ scripts/bootstrap-ansible.sh
```

Configurer le déploiement:

OpenStack-Ansible (OSA) dépend de divers fichiers utilisés pour créer un inventaire pour Ansible.

Effectuez la configuration suivante sur l'hôte de déploiement.

```
# Controller1 #
$ cp -r /opt/openstack-ansible/etc/openstack_deploy /etc/openstack_deploy
$ cd /etc/openstack_deploy
$ vim openstack_user_config.yml
```

Ajoutez les configurations suivantes au fichier:

```
---
cidr_networks:
  container: 172.29.236.0/22
  tunnel: 172.29.240.0/22
  storage: 172.29.244.0/22

used_ips:
- "172.29.236.1,172.29.236.50"
- "172.29.240.1,172.29.240.50"
- "172.29.244.1,172.29.244.50"

global_overrides:
  # The internal and external VIP should be different IPs, however they
  # do not need to be on separate networks.
  external_lb_vip_address: 10.62.1.1
  internal_lb_vip_address: 172.29.236.10
  management_bridge: "br-mgmt"
  provider_networks:
    - network:
        container_bridge: "br-mgmt"
        container_type: "veth"
        container_interface: "eth1"
        ip_from_q: "container"
```

```
type: "raw"
group_binds:
  - all_containers
  - hosts
is_container_address: true
- network:
  container_bridge: "br-vxlan"
  container_type: "veth"
  container_interface: "eth10"
  ip_from_q: "tunnel"
  type: "vxlan"
  range: "1:1000"
  net_name: "vxlan"
  group_binds:
    - neutron_linuxbridge_agent
- network:
  container_bridge: "br-vlan"
  container_type: "veth"
  container_interface: "eth12"
  host_bind_override: "eth12"
  type: "flat"
  net_name: "flat"
  group_binds:
    - neutron_linuxbridge_agent
- network:
  container_bridge: "br-vlan"
  container_type: "veth"
  container_interface: "eth11"
  type: "vlan"
  range: "101:200,301:400"
  net_name: "vlan"
  group_binds:
    - neutron_linuxbridge_agent
- network:
  container_bridge: "br-storage"
  container_type: "veth"
  container_interface: "eth2"
  ip_from_q: "storage"
  type: "raw"
  group_binds:
    - glance_api
    - cinder_api
    - cinder_volume
    - nova_compute
```

```
###
### Infrastructure
###

# galera, memcache, rabbitmq, utility
shared-infra_hosts:
  infra1:
    ip: 172.29.236.10

# repository (apt cache, python packages, etc)
repo-infra_hosts:
  infra1:
    ip: 172.29.236.10

# load balancer
haproxy_hosts:
  infra1:
    ip: 172.29.236.10

###
### OpenStack
###

# keystone
identity_hosts:
  infra1:
    ip: 172.29.236.10

# cinder api services
storage-infra_hosts:
  infra1:
    ip: 172.29.236.10

# glance
image_hosts:
  infra1:
    ip: 172.29.236.10

# placement
placement-infra_hosts:
  infra1:
    ip: 172.29.236.10

# nova api, conductor, etc services
compute-infra_hosts:
  infra1:
    ip: 172.29.236.10
```

```
# heat
orchestration_hosts:
  infra1:
    ip: 172.29.236.10

# horizon
dashboard_hosts:
  infra1:
    ip: 172.29.236.10

# neutron server, agents (L3, etc)
network_hosts:
  infra1:
    ip: 172.29.236.10

# nova hypervisors
compute_hosts:
  compute1:
    ip: 172.29.236.11

# cinder storage host (LVM-backed)
storage_hosts:
  storage1:
    ip: 172.29.236.13
    container_vars:
      cinder_backends:
        limit_container_types: cinder_volume
      lvm:
        volume_group: cinder-volumes
        volume_driver: cinder.volume.drivers.lvm.LVMVolumeDriver
        volume_backend_name: LVM_iSCSI
        iscsi_ip_address: "172.29.244.13"
```

```
# Controller1 #
$ vim /etc/openstack_deploy/user_variables.yml
```

Ajoutez les configurations suivantes au fichier:

```
---
# This file contains an example of the global variable overrides
# which may need to be set for a production environment.

## OpenStack public endpoint protocol
openstack_service_publicuri_proto: http
```

Nous utiliserons le script **pw-token-gen.py** pour générer des valeurs aléatoires pour les variables d'identification de chaque service.

```
# Controller1 #
$ cd /opt/openstack-ansible
$ ./scripts/pw-token-gen.py --file /etc/openstack_deploy/user_secrets.yml
```

Exécuter des playbooks:

Le processus d'installation nécessite l'exécution de trois playbooks principaux :

- ❖ Le playbook **setup-hosts.yml**: prépare les hôtes cibles pour l'infrastructure et les services OpenStack, construit et redémarre des conteneurs sur des hôtes cibles et installe des composants communs dans des conteneurs sur des hôtes cibles.
- ❖ Le playbook **setup-infrastructure.yml**: installe les services d'infrastructure : Memcached, le serveur de référentiel, Galera, RabbitMQ et rsyslog.
- ❖ Le playbook **setup-openstack.yml**: installe les services OpenStack, y compris Identity (keystone), Image (glance), Block Storage (cinder), Compute (nova), Networking (neutron), etc.

```
# Controller1 #
$ cd /opt/openstack-ansible/playbooks
$ openstack-ansible setup-infrastructure.yml --syntax-check
```

vous devriez voir un résultat comme celui-ci contient le succès de l'exécution:

```
root@leet:/home/leet# cd /opt/openstack-ansible/playbooks/
root@leet:/opt/openstack-ansible/playbooks# openstack-ansible setup-infrastructure.yml --syntax-check
Variable files: "-e @/etc/openstack_deploy/user_secrets.yml -e @/etc/openstack_deploy/user_variables.yml "
[WARNING]: Unable to parse /etc/openstack_deploy/inventory.ini as an inventory
source
[DEPRECATION WARNING]: "include" is deprecated, use include_tasks/import_tasks
instead. This feature will be removed in version 2.16. Deprecation warnings can
be disabled by setting deprecation_warnings=False in ansible.cfg.

playbook: setup-infrastructure.yml

EXIT NOTICE [Playbook execution success] *****
=====
```

```
# Controller1 #
$ openstack-ansible setup-hosts.yml
$ openstack-ansible setup-infrastructure.yml
```

Exécutez la commande suivante pour vérifier le cluster de bases de données :

```
# Controller1 #
$ ansible galera_container -m shell -a "mysql -h localhost -e 'show status
like \"%wsrep_cluster_%\";'"
```

vous devriez voir un résultat comme celui-ci:

```
root@leet:/opt/openstack-ansible/playbooks# ansible galera_container -m shell \
> -a "mysql -h localhost -e 'show status like \"%wsrep_cluster_%\";'"
Variable files: "-e @/etc/openstack_deploy/user_secrets.yml -e @/etc/openstack_deploy/user_variables.yml "
[WARNING]: Unable to parse /etc/openstack_deploy/inventory.ini as an inventory
source
infra1_galera_container-a06e56c0 | CHANGED | rc=0 >>
Variable_name  Value
wsrep_cluster_weight      1
wsrep_cluster_capabilities
wsrep_cluster_conf_id    1
wsrep_cluster_size       1
wsrep_cluster_state_uuid  85b3441b-dd42-11ec-b6e8-66b43055e747
wsrep_cluster_status      Primary
```

```
# Controller1 #
$ openstack-ansible setup-openstack.yml
```

chaque fois que vous exécutez un playbook, vous devriez voir un résultat comme celui-ci qui s'est terminé avec zéro élément inaccessible ou échoué :

```
changed: [infra1_utility_container-05b1c6a5]

RUNNING HANDLER [python_venv_build : venv changed] *****

PLAY RECAP *****
compute1          : ok=203  changed=86  unreachable=0    failed=0    skipped=56  rescued=0
                   ignored=0
infra1            : ok=84   changed=38  unreachable=0    failed=0    skipped=19  rescued=0
                   ignored=0
infra1_cinder_api_container-65af3684 : ok=158  changed=67  unreachable=0    failed=0    skipped=37  rescued=0
                   ignored=0
infra1_glance_container-d800da97 : ok=122  changed=60  unreachable=0    failed=0    skipped=19  rescued=0
                   ignored=0
infra1_heat_api_container-901c8d5c : ok=130  changed=69  unreachable=0    failed=0    skipped=27  rescued=0
                   ignored=0
infra1_horizon_container-19adba6e : ok=79   changed=44  unreachable=0    failed=0    skipped=9   rescued=0
                   ignored=0
infra1_keystone_container-ddcff8fc : ok=182  changed=94  unreachable=0    failed=0    skipped=39  rescued=0
                   ignored=0
infra1_neutron_server_container-1aa98108 : ok=100  changed=54  unreachable=0    failed=0    skipped=18  rescued=0
                   ignored=0
infra1_nova_api_container-15365bd1 : ok=175  changed=81  unreachable=0    failed=0    skipped=47  rescued=0
                   ignored=0
infra1_placement_container-69a5fbbe : ok=101  changed=51  unreachable=0    failed=0    skipped=17  rescued=0
                   ignored=0
infra1_utility_container-05b1c6a5 : ok=33   changed=16  unreachable=0    failed=0    skipped=6   rescued=0
                   ignored=0
localhost         : ok=3    changed=3   unreachable=0    failed=0    skipped=1   rescued=0
                   ignored=0
storage1          : ok=97   changed=45  unreachable=0    failed=0    skipped=37  rescued=0
                   ignored=0

EXIT NOTICE [Playbook execution success] *****
=====
```

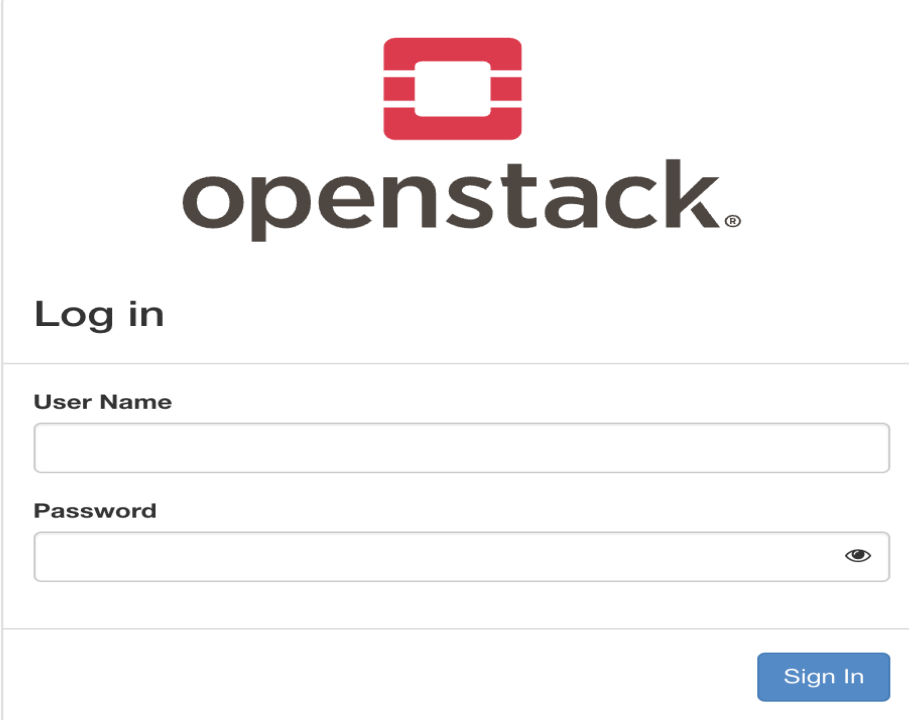

Vérification du fonctionnement d'OpenStack:

```
# Controller1 #
$ lxc-attach -n `lxc-ls -1 | grep utility | head -n 1`
$ source ~/openrc
$ nova hypervisor-list
```

vous devriez voir un résultat comme celui-ci:

```
root@infra1-utility-container-1466467f:/# nova hypervisor-list
+-----+-----+-----+-----+
| ID                | Hypervisor hostname | State | Status |
+-----+-----+-----+-----+
| 92a18065-fa48-4cff-832c-f8872f746661 | leet                | up    | enabled |
+-----+-----+-----+-----+
root@infra1-utility-container-1466467f:/#
```

Avec un navigateur web, accédez au Dashboard en utilisant l'url : <https://10.62.1.1/>



The image shows the OpenStack login interface. At the top is the OpenStack logo, which consists of a red square with a white 'O' inside, followed by the word 'openstack' in a bold, black, sans-serif font. Below the logo is the text 'Log in'. Underneath, there are two input fields: 'User Name' and 'Password'. The 'Password' field has a small eye icon to its right, indicating a toggle for password visibility. At the bottom right of the form is a blue button with the text 'Sign In'.

pour obtenir le mot de passe administrateur exécutez la commande suivante:

```
# Controller1 #
$ cat /etc/openstack_deploy/user_secrets.yml | grep
keystone_auth_admin_password
```

Le nom d'utilisateur est : **admin**

Résoudre les problèmes:

Si vous ne trouvez pas d'hyperviseur vérifiez que votre nom d'hôte ne commence pas par des chiffres, si c'est le cas changez définitivement votre nom d'hôte pour un autre qui ne commence pas par des chiffres.

```
May 26 21:01:44 1337 nova-compute[18013]: ValueError: 1337 contains no non-numeric characters in the top-level domain part of the host name and
May 26 21:01:44 1337 nova-compute[18013]: During handling of the above exception, another exception occurred:
May 26 21:01:44 1337 nova-compute[18013]: Traceback (most recent call last):
May 26 21:01:44 1337 nova-compute[18013]:   File "/openstack/venvs/nova-24.1.0.dev184/lib/python3.8/site-packages/oslo_config/types.py", line 8>
May 26 21:01:44 1337 nova-compute[18013]:     value = super(HostDomain, self).__call__(value)
May 26 21:01:44 1337 nova-compute[18013]:   File "/openstack/venvs/nova-24.1.0.dev184/lib/python3.8/site-packages/oslo_config/types.py", line 8>
May 26 21:01:44 1337 nova-compute[18013]:     raise ValueError(
May 26 21:01:44 1337 nova-compute[18013]: ValueError: 1337 is not a valid host address
```

```
# Controller1 & Compute1 & Storage1 #
$ hostname
$ hostnamectl set-hostname NEWHOSTNAME
$ vim /etc/hosts
# Remplacez toute occurrence du nom d'hôte existant par votre nouveau
$ reboot
```

Si vous ne pouvez pas créer d'instance, vérifiez si le module du noyau kvm est chargé dans **Compute1** avec la commande suivante :

```
# Compute1 #
$ lsmod | grep kvm
```

vous devriez voir un résultat comme celui-ci:

```
root@leet:~# lsmod | grep kvm
kvm_intel                282624    12
kvm                      663552    1 kvm_intel
root@leet:~#
```

sinon, vous devez le démarrer avec les commandes suivantes:

```
# Compute1 #
$ modprobe kvm_intel
$ modprobe kvm
$ echo kvm_intel >> /etc/modules
$ echo kvm >> /etc/modules
```