# Mise en place d'une VM avec 2 interfaces réseaux sur OVH Cloud Public:

## Deployer un réseau privé

```
openstack network create net-private-test
openstack subnet create sub-private-test --network net-private-test --subnet-range 192.168.1.0/24 --gateway none
```

## Créer Key pair ssh

```
openstack keypair create private-test
```

## deployer VM1 avec une interface sur le réseau privé + sur le réseau public

```
openstack server create VM1 --image "Ubuntu 18.04" --flavor "s1-2" --key-name "testkey" --net "Ext-Net" --net "net-private-test"
```

- Option secondaire: créer VM avec une Interface Public et ensuite attacher un port sur le private

```
  `openstack server create VM1 --image "Ubuntu 18.04" --flavor "s1-2" --key-name "testkey" --net "Ext-Net"  `
  `openstack port create --network net-private-test private-test`
  `openstack server add port VM1 private-test`
```

  - Configuration de l'interface private:
    récupérer le nom de l'interface down
    `ip a s | grep DOWN | awk -F ":" '{print $2}'`
    récupérer l'@ mac de l'interface down
    `ip a s | grep DOWN -A1 | awk -F " " '{print $2}' | grep -v ens`
    blqoue à ajouter à la fin du fichier /etc/netplan/50-cloud-init.yaml

```
        ethernets:
            ens4:
                dhcp4: true
                match:
                    macaddress: fa:16:3e:c9:8b:fe
```

script auto enable private int:

```
#!/bin/bash
```

```
INT_DOWN=`ip a s | grep DOWN | awk -F ":" '{print $2}'`
MAC_ADDR=`ip a s | grep $INT_DOWN -A1 | awk -F " " '{print $2}' | grep -v
$INT_DOWN`
NETPLAN_FILE="/etc/netplan/50-cloud-init.yaml"
echo $MAC_ADDR

INT_CONF='    ethernets:\n
                $INT_DOWN:\n
                    dhcp4: true\n
                    match:\n
                        macaddress: $MAC_ADDR'
echo "    ethernets:" >> $NETPLAN_FILE
echo "        $INT_DOWN:" >> $NETPLAN_FILE
echo "            dhcp4: true" >> $NETPLAN_FILE
echo "            match:" >> $NETPLAN_FILE
echo "                macaddress: $MAC_ADDR" >> $NETPLAN_FILE

`netplan apply`
```

il est possible de faire excuter le script par openstack au démarrage de la vm

```
openstack server create VM1 --image "Ubuntu 18.04" --flavor "s1-2" --key-name
"private-test-thales" --net "Ext-Net" --net "net-private-test" --user-data
'test.sh'
```

### Test une connexion SSH VM1

```
ok
```

### Déployer VM2 avec une interface sur le réseau privé

```
openstack server create VM2 --image "Ubuntu 18.04" --flavor "s1-2" --key-name
"testkey" --net "net-private-test"
```

### test la communication entre VM1 et VM2

```
ok mais la VM n'a pas accès à internet
```

# Mise en place d'un proxy squid

sur vm 1 installation et configuration de squid pour authoriser l'accès à internet au machine du sous réseau

- https://www.it-connect.fr/mise-en-place-et-configuration-dun-proxy-avec-squid/
- https://doc.ubuntu-fr.org/proxy_terminal

Installation de Squid

```
    apt-get install squid
    systemctl start squid
    systemctl enable squid
    mv /etc/squid/squid.conf /etc/squid/squid.conf.bak
    vim /etc/squid/squid.conf
```

Fichier de configuration de squid:

```
# Squid a besoin de savoir le nom de la machine, notre machine s'appelle
srv-proxy, donc :
visible_hostname srv-proxy

# Par défaut le proxy écoute sur ses deux interfaces, pour des soucis de
sécurité il faut donc le
# restreindre à écouter sur l'interface du réseau local (LAN)
http_port 192.168.1.56:3128

# Changer la taille du cache de squid, changer la valeur 100 par ce que
vous voulez (valeur en Mo)
cache_dir ufs /var/spool/squid 100 16 256
acl all src all # ACL pour autoriser/refuser tous les réseaux (Source =
All) - ACL obligatoire
acl lan src 192.168.1.0/24 # ACL pour autoriser/refuser le réseau
192.168.1.0/
acl Safe_ports port 80 # Port HTTP = Port 'sure'
acl Safe_ports port 443 # Port HTTPS = Port 'sure'
acl Safe_ports port 21 # Port FTP = Port 'sure'
acl Safe_ports port 22 # Port FTP = Port 'sure'
#############################################################################
##

# Désactiver tous les protocoles sauf les ports sures
http_access deny !Safe_ports
# Désactiver l'accès pour tous les réseaux sauf les clients de l'ACL Lan
# deny = refuser ; ! = sauf ; lan = nom de l'ACL à laquelle on fait
référence.
http_access deny !lan

# Port utilisé par le Proxy :
# Le port indiqué ici, devra être celui qui est précisé dans votre
navigateur.
http_port 3128
```

redémarage du squid pour prise en compte de la config

```
systemctl restart squid
```

sur vm 2 ajout de la variable d'environnement http_proxy:

```
export http_proxy=http://192.168.1.56:3128
```

==> la VM2 a maintenant accès à internet.