



OpenStack pour les développeurs

Jérôme PANSANEL

jerome.pansanel@iphc.cnrs.fr

Formation Utilisateur FG-Cloud – Avril 2016 – Lyon



Pré-requis

Interpréteur Python v2.7 (ou 2.6 avec les anciennes versions)

Les paquets suivants :

- python-cinderclient
- python-glanceclient
- python-keystoneclient
- python-neutronclient
- python-novaclient

Démarrer avec l'API OpenStack

Mise en place d'environnement :

```
#!/usr/bin/env python
import os

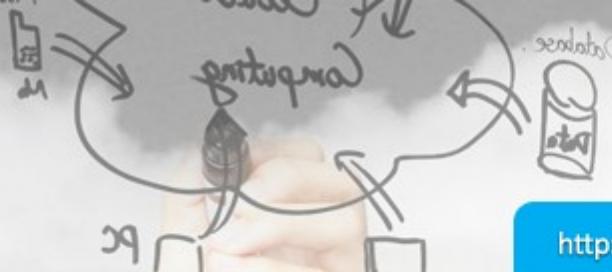
from novaclient.v1_1 import client

def get_nova_creds():
    creds = {}
    creds['username'] = os.environ['OS_USERNAME']
    creds['api_key'] = os.environ['OS_PASSWORD']
    creds['project_id'] = os.environ['OS_TENANT_NAME']
    creds['auth_url'] = os.environ['OS_AUTH_URL']
    return creds

creds = get_nova_creds()

nova = client.Client(**creds)

nova.client.verify_cert = False
```



Récupérer les paramètres du Cloud

```
>>> nova.flavors.list()
```

```
[<Flavor: m1.tiny>, <Flavor: m1.small>, <Flavor: m1.medium>, <Flavor: m1.large>, <Flavor: m1.xlarge>, <Flavor: m1.2xlarge>, <Flavor: m1.cms-small>, <Flavor: m1.small-bigmem>]
```

```
>>> nova.images.list()
```

```
[<Image: docker_FG>, <Image: CentOS-7-x86_64-Minimal-ISO-Installer>, <Image: centos7_formation>, <Image: FG CentOS 7 x86_64 Generic Cloud Image>, <Image: Image for CentOS 6 minimal [CentOS/6.x/KVM]_Appliance>, <Image: Image for TinyCoreLinux [Other/TinyCoreLinux/QEMU-KVM]_Appliance>, <Image: CirrOS>]
```

```
>>> nova.networks.list()
```

```
[<Network: ext-net>, <Network: fg-formation-net>]
```



Démarrer une VM

```
>>> image = nova.images.list()[3]

>>> image.id
u'74f127bc-d294-45ca-ab19-63fd9add5e9'

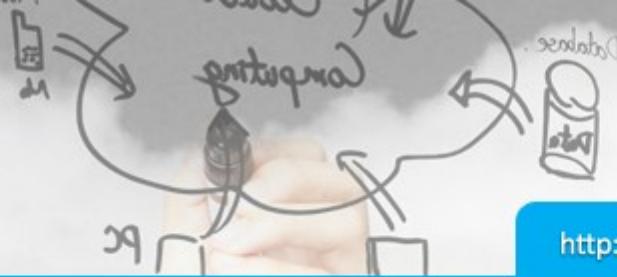
>>> flavor = nova.flavors.list()[1]

>>> flavor.id
u'2'

>>> network = nova.networks.list()[1]

>>> network.id
u'2c36d255-01ce-4330-93e1-13f8d2cec7fd'

>>> server = nova.servers.create(name = 'fg_formationX',
                                   image = image.id,
                                   flavor = flavor.id,
                                   network = network.id,
                                   key_name = 'cloudkey')
```



Démarrer une VM

```
>>> server.status  
u'BUILD'
```

```
>>> server.status  
U'BUILD'
```

```
>>> server.get()
```

```
>>> server.status  
u'ACTIVE'
```



Gestion du réseau

```
>>> server.addresses
```

```
{u'fg-formation-net': [{u'OS-EXT-IPS-MAC:mac_addr': u'fa:16:3e:3e:95:2c',  
u'version': 4, u'addr': u'172.16.7.144', u'OS-EXT-IPS:type': u'fixed'}] }
```

```
>>> nova.floating_ip_pools.list()
```

```
[<FloatingIPPool: name=ext-net>]
```

```
>>> nova.floating_ips.list()
```

```
[<FloatingIP fixed_ip=172.16.7.142, id=10e6c828-1516-41a4-b6c8-  
3c9414cf97e3, instance_id=4e5fd97a-899f-4d06-ac5-9e3f30e3a8c1,  
ip=134.158.151.49, pool=ext-net>, <FloatingIP fixed_ip=None, id=f38ad3f2-  
7d51-4488-b213-217800f704ec, instance_id=None, ip=134.158.151.56, pool=ext-  
net>, ...]
```

```
>>> for floating_ip in nova.floating_ips.list():
```

```
...     if not floating_ip.instance_id:
```

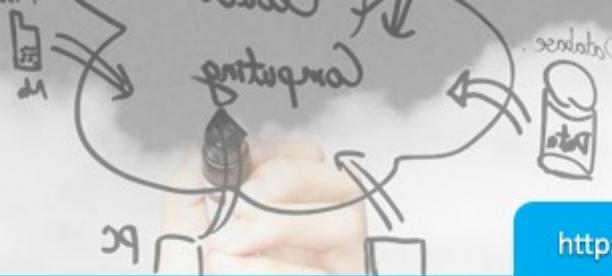
```
...         print(floating_ip.ip)
```

```
134.158.151.238
```

```
134.158.151.35
```

```
134.158.151.85
```

```
...
```



Gestion du réseau

```
>>> floating_ip = nova.floating_ips.list()[5]

>>> server.add_floating_ip('134.158.151.35')

>>> server.get()

>>> server.addresses
{u'fg-formation-net': [{u'OS-EXT-IPS-MAC:mac_addr': u'fa:16:3e:3e:95:2c',
u'version': 4, u'addr': u'172.16.7.144', u'OS-EXT-IPS:type': u'fixed'}, {u'OS-
EXT-IPS-MAC:mac_addr': u'fa:16:3e:3e:95:2c', u'version': 4, u'addr':
u'134.158.151.35', u'OS-EXT-IPS:type': u'floating'}]}
```



Gestion du stockage permanent

```
>>> from cinderclient.v2 import client as ciclient  
  
>>> cinder = ciclient.Client(**creds)  
  
>>> cinder.client.verify_cert = False  
  
>>> cinder.volumes.list()  
[<Volume: f0180abe-3962-4195-9581-04f97c3775de>, <Volume: 2872fd72-  
bd27-42ac-94e9-38786392918c>, <Volume: 246c6ccc-c176-4fcb-aa8f-  
182b3399e47f>, <Volume: b7511712-9fc3-447e-988d-27aedb1a31>,  
<Volume: e81db57c-78fb-4e16-a3e9-3f717181fb3c>, <Volume: 7b4ec973-  
f181-44bf-b7f9-99cab37faaead>, <Volume: cefa719f-c05f-4925-bb8e-  
846f5fd53639>]  
  
>>> volume = cinder.volumes.create(1,  
                                     name='stockage_testX',  
                                     description='un stockage de test')  
  
>>> volume  
<Volume: e12da5bc-8fdc-43c7-8a23-df486d2e2057>
```

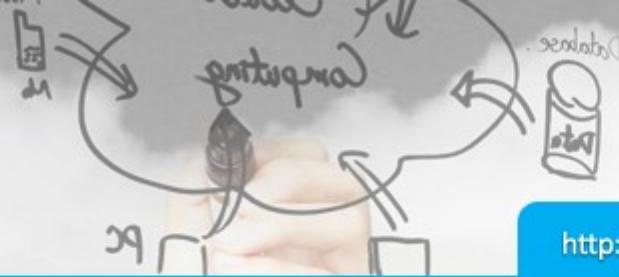


Gestion du stockage permanent

```
>>> nova.volumes.create_server_volume(server.id,volume.id, None)
<Volume: e12da5bc-8fdc-43c7-8a23-df486d2e2057>
```

```
>>> server.get()
```

```
>>> server.to_dict()[u'os-extended-volumes:volumes_attached']
[{'id': u'e12da5bc-8fdc-43c7-8a23-df486d2e2057'}]
```



Finir une session

1. Détacher le volume

```
>>> nova.volumes.delete_server_volume(server.id,volume.id)
```

2. Arrêter la machine manuellement

3. Redémarrer la machine ou supprimer la VM

```
# reboot_type can be "SOFT" or "HARD"
```

```
server.reboot(reboot_type)
```

```
# Suppression de la VM
```

```
server.delete()
```



APIs OpenStack : documentation

- Documentation IBM :

<http://www.ibm.com/developerworks/cloud/library/cl-openstack-pythonapis/>

- Guide de démarrage rapide :

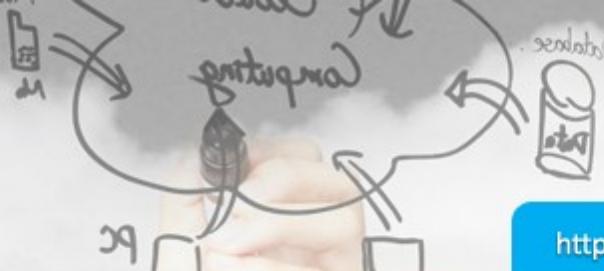
<http://docs.openstack.org/api/quick-start/content/>

- Guide de référence :

<http://developer.openstack.org/api-ref-compute-v2.1.html>

- Documentation en français :

<http://dev.cloudwatt.com/fr/doc/sdk/sdk-python.html>



Questions ?