Cloud, Cluster and Grid Computing

Private cloud deployment

Author: Mihai CARABAȘ

Editura POLITEHNICA PRESS

Reviewers:

Răzvan RUGHINIȘ

Sergiu WIESZ

ISBN: 978-606-515- 955-6

Contents

Private cloud infrastructure: Openstack

- The tasks in this lab will done in the faculty's OpenStack cloud. We will create, modify and delete different cloud objects (instances, networks, subnets etc.)
- For interaction with OpenStack, we will use the official OpenStack clients (`nova`, `neutron`, `keystone` etc.)
- All the clients are already installed on `fep.grid.pub.ro`, so we'll use them from there.

Authentication

Before using the clients, we must provide the necessary authentication parameters. This is done via an OpenStack RC file. To obtain your OpenStack RC from Horizon (the OpenStack dashboard), go to **Project → API Access** and click on **Download OpenStack RC file (Identity API v3)**.
Upload the file in your home directory on `fep.grid.pub.ro` and source it in Bash:

```
$ source alexandru.carp_prj-openrc.sh

Please enter your OpenStack Password:
```

Enter your password, and for verifying that authentication is correct, enter any OpenStack command. For example, list the catalog of installed services, using **openstack catalog list**:

```
$ openstack catalog list

+---------------+----------------+-------------------------------------------
-----------------------------------------------+

|   Name                        |   Type                              |   Endpoints
|

+---------------+----------------+-------------------------------------------
-----------------------------------------------+

|   designate                |   dns                                      |   NCIT
|

|                 |                  |   publicURL: http://172.16.5.161:9001/v1
|

|                |                  |   internalURL: http://172.16.5.161:9001/v1
|

|                 |                  |   adminURL: http://172.16.5.161:9001/v1
|

|                                    |                                          |
|
```

```
|   nova                              |   compute                          |   NCIT
|

|                        |                          |    publicURL: http://cloud-
controller.grid.pub.ro:8774/v2/e81c0aa57f61461c8da8496157f7041e   |

|                        |                          |    internalURL: http://cloud-
controller.grid.pub.ro:8774/v2/e81c0aa57f61461c8da8496157f7041e |

|                        |                          |    adminURL: http://cloud-
controller.grid.pub.ro:8774/v2/e81c0aa57f61461c8da8496157f7041e    |

|                                     |                                                |
|

[...]
```

Listing resources

For booting an instance, we must know the following parameters (objects):

- **image** - the name or ID of the image used to boot the instance
- **flavor** - the name of ID of the flavor (which specifies the virtual resources size: CPU, RAM, Disk)
- **keypair** - which SSH keypair to inject into the instance at first boot
- **network** - to which network to attach the vNIC of the instance
- **security-group** - which security group (set of filtering rules) to apply to the network port

For each of the above objects, we will list what is available in our own OpenStack tenant.

Images

Images are handled by the **Glance** client. We will list them using **glance image-list**:

```
$ glance image-list

+--------------------------------------+-----------------------------------
--------+

| ID                                   | Name                                |

+--------------------------------------+-----------------------------------
--------+

|   53fec0b8-753e-4a4f-91a3-51624a8a270d   |   ABD   Oracle   Template   v1
|

| 6349c723-de9e-4d3a-900b-b07416e5e486 | ABD Template v3                     |
```

```
|    56f8b431-d7be-43d1-966e-9c974fe20c8f    |    ASCG/CCG    Template    v2
|

| 1aa5c205-9dde-4382-b7d7-8b5d652e38b8 | Centos 6                              |

| c3dd305d-84e1-4e72-9a15-8194a4aafef3 | Centos 7                              |

|     43259067-c2f1-438f-a16a-c4ad86dc2ad2     |     Cisco     onePK-1.3.0.181
|

|    6a9c3513-d5d6-48f8-9600-e6afe9ac6686    |    Cloudera    Hadoop    5.7.1
|

| 692e7272-9961-48d2-b54f-9f7cf3a45262 | Debian 8.6.0                          |

|   f9a48d01-9123-4fb1-b989-908ac414a339   |   GSR   Template   (Debian   8.6.0)
|

| 4a69c6e9-fe89-409f-931a-41c75c12de3c | IBM-SDP                               |

|     c667704b-d650-4f8c-b08f-ba05583d8428     |     ISC     Temaplate     v2.1
|

| aa935990-0751-40f6-b40a-c9e9017e939e | ISC Template v2                       |

| 9481ec27-6897-4c3e-87ec-48977cb66164 | ISRM Template v2                      |

| 0d82a5c3-1141-45f0-83c3-e6c8157fc511 | Openstack Juno                        |

| ad74e12d-e883-4338-a1e6-78d2e0eb3a24 | RL 2016                               |

| c3e91d38-0e7e-4506-a922-c4911bc9fca9 | RL 2016 Tema2                         |

| 31b91232-d87c-44ba-8abe-d99580a6375f | SCGC Template v1                      |

| 7c01be41-0973-4cec-b00f-9c7116c9f885 | SO VM Linux v1                        |

| c724142c-75ee-45b1-9f47-b752011d9bbc | STR - Win7                            |

|     3672370a-af54-47c2-b2c1-d9875952415f     |     Ubuntu     16.04     Xenial
|

|   7aae1675-571f-487b-9526-14a1ae038bc3   |   Ubuntu   16.04   Xenial   (32bit)
|

| 7f283e7e-b347-408b-8521-daeec831b456 | USO Practic Template (Ubuntu 16.04 -
32bit) |

| 2eeb7d33-fc5a-4f71-b3ae-bfef6b1ce9dd | USO Template (Ubuntu 16.04 - 32bit)
|

| 9885d828-78d2-4804-b816-b7072aa4e08a | WinXP SCPI                            |

+--------------------------------------+--------------------------------------
--------+
```

For booting the instance, we will use the **Ubuntu 16.04 Xenial** image, which has the ID **3672370a-af54-47c2-b2c1-d9875952415f**. Let's find some more information about this image using **glance image-show**:

```
$ glance image-show 3672370a-af54-47c2-b2c1-d9875952415f

+------------------+--------------------------------------+

| Property         | Value                                |

+------------------+--------------------------------------+

| checksum         | 02f5162d90e1a620177c3075266f734b     |

| container_format | bare                                 |

| created_at       | 2016-10-24T17:50:20Z                 |

| disk_format      | qcow2                                |

| id               | 3672370a-af54-47c2-b2c1-d9875952415f |

| min_disk         | 0                                    |

| min_ram          | 0                                    |

| name             | Ubuntu 16.04 Xenial                  |

| owner            | 1836fc3aec3f4226a73bb5e249385fe0     |

| protected        | True                                 |

| size             | 313982976                            |

| status           | active                               |

| tags             | []                                   |

| updated_at       | 2016-10-29T13:26:14Z                 |

| virtual_size     | None                                 |

| visibility       | public                               |

+------------------+--------------------------------------+
```

Flavors

Flavors are handled in **Nova** (the compute service). We will list them using **nova flavor-list**:

```
$ nova flavor-list

+--------------------------------------+-----------+-----------+------+------
-----+------+-------+-------------+-----------+
```

```
| ID                                   | Name      | Memory_MB | Disk | Ephemeral
| Swap | VCPUs | RXTX_Factor | Is_Public |

+--------------------------------------+-----------+-----------+------+------
-----+------+-------+-------------+-----------+

| 3c183fea-cea6-489e-8b6e-d34c4bf073ec | m1.tiny    | 512         | 8     | 0
|      | 1     | 1.0         | True      |

| 3e708b91-53c8-436c-9f6f-653f9a403481 | m1.medium | 1536        | 16    | 0
|      | 1     | 1.0         | True      |

| 443d714c-f295-4c92-b75e-96ae99a64fc4 | m1.large   | 4096        | 10    | 0
|      | 2     | 1.0         | True      |

| 4d76ded7-fae0-4fd4-9191-600a466a5fea | c1.large   | 4096        | 16    | 0
|      | 4     | 1.0         | True      |

| 5b1624ef-30b3-4eba-bfe4-0a1dc5211594 | c1.small   | 1024        | 16    | 0
|      | 1     | 1.0         | True      |

| 77c51a45-3f34-45e7-b9d9-84ef1266be83 | c1.medium | 1536        | 16    | 0
|      | 2     | 1.0         | True      |

| a8578051-c828-4446-97d0-da34b8877348 | m1.xlarge | 4096        | 24    | 0
| 2048 | 4     | 1.0         | True      |

| d1ddad6d-3a87-4f4f-a460-ec2c324d42b7 | m1.small   | 1024        | 10    | 0
|      | 1     | 1.0         | True      |

+--------------------------------------+-----------+-----------+------+------
-----+------+-------+-------------+-----------+
```

Let's find more information about the **m1.tiny** flavor, having ID **3c183fea-cea6-489e-8b6e-d34c4bf073ec**, using **nova flavor-show**:

```
$ nova flavor-show 3c183fea-cea6-489e-8b6e-d34c4bf073ec

+----------------------------+--------------------------------------+

| Property                   | Value                                |

+----------------------------+--------------------------------------+

| OS-FLV-DISABLED:disabled   | False                                |

| OS-FLV-EXT-DATA:ephemeral  | 0                                    |

| disk                       | 8                                    |

| extra_specs                | {"type": "gp"}                       |

| id                         | 3c183fea-cea6-489e-8b6e-d34c4bf073ec |

| name                       | m1.tiny                              |
```

```
| os-flavor-access:is_public | True                                 |
| ram                        | 512                                  |
| rxtx_factor                | 1.0                                  |
| swap                       |                                      |
| vcpus                      | 1                                    |
+----------------------------+--------------------------------------+
```

Keypairs

Keypair are also handled by **Nova**. To list them, we use **nova keypair-list**. You should only see your own keypair(s):

```
$ nova keypair-list
+------+-------------------------------------------------+
| Name | Fingerprint                                     |
+------+-------------------------------------------------+
| fep  | ee:ed:db:ae:e3:09:f9:a0:f2:6f:4f:47:4a:15:14:e4 |
+------+-------------------------------------------------+
```

Using **nova keypair-show**, you can see the details, including the public key:

```
$ nova keypair-show fep
+-------------+-------------------------------------------------+
| Property    | Value                                           |
+-------------+-------------------------------------------------+
| created_at  | 2017-02-27T09:06:16.000000                      |
| deleted     | False                                           |
| deleted_at  | -                                               |
| fingerprint | ee:ed:db:ae:e3:09:f9:a0:f2:6f:4f:47:4a:15:14:e4 |
| id          | 2433                                            |
| name        | fep                                             |
| updated_at  | -                                               |
| user_id     | alexandru.carp                                  |
```

```
+-------------+-----------------------------------------------------+

Public                                key:                                    ssh-rsa
AAAAB3NzaC1yc2EAAAADAQABAAABAQCqnHjfEFfB6n6CbF5a4wQpnxZkavrJCuX1ivjNoGjUmEa9d
n0GS+YB+bWs8Nny8cgNkgzRE1jFcIZ2ByNxahf884G2QNZm+9tufWl3V0GdqZ+sooi5Fry9BGv/DH
yRw3/y+w9xSfOoS8pFl/lV3jOfZYEWLRTwVT63SOx1sjOOMJtBxr6IyjHzVWErKlJymuxa7R5u4Yq
qeCNpqNYCQZqvbbY6iM9Rd4WbmEQgTtpmM6TLE2mpaD9MTeFsoiQxPhCTGCr1EZsLtIdPkowMAxCv
VEQ7GU7p4R/8WZtKpujNLboFyZkZm7Ku0JgNdNEc+sl5YzxS9E6BkxlIl1xefEFP
alexandru.carp@fep7-1.grid.pub.ro
```

Networks

Networks are handled by **Neutron**. We use **neutron net-list** to list all the networks:

```
$ neutron net-list

+--------------------------------------+--------+--------------------------
----------------------+

|  id                                                  |  name     |  subnets
|

+--------------------------------------+--------+--------------------------
----------------------+

|  fc56b8a7-d6ea-4025-8fba-0f868499e20b  |  Net224  |  ec56c500-2508-46d9-b2bd-
a601cd7c1565             |

|  525c2933-8a11-4cfc-ad12-234cac7c9328  |  Net240  |  f42d03c0-d5e4-4769-b655-
1a59144b01e5             |

|  424666ed-c0e8-4d1c-96fe-c22c56262a87  |  vlan9   |  3f7ca0ff-7855-4f12-b6b4-
4c4a763aa22f 10.9.0.0/16 |

+--------------------------------------+--------+--------------------------
----------------------+
```

Let's show details about:

- the **vlan9** network, with ID **424666ed-c0e8-4d1c-96fe-c22c56262a87** (using **neutron net-show**)
- its associated subnet, with ID **3f7ca0ff-7855-4f12-b6b4-4c4a763aa22f** (using **neutron subnet-show**)

```
$ neutron net-show 424666ed-c0e8-4d1c-96fe-c22c56262a87

+-----------------+--------------------------------------+

| Field           | Value                                |

+-----------------+--------------------------------------+
```

```
| admin_state_up  | True                                 |
| id              | 424666ed-c0e8-4d1c-96fe-c22c56262a87 |
| mtu             |                                      |
| name            | vlan9                                |
| router:external | True                                 |
| shared          | True                                 |
| status          | ACTIVE                               |
| subnets         | 3f7ca0ff-7855-4f12-b6b4-4c4a763aa22f |
| tenant_id       | 975459f77464498898a1b17c8f08c8d4     |
+-----------------+--------------------------------------+
$ neutron subnet-show 3f7ca0ff-7855-4f12-b6b4-4c4a763aa22f
+-------------------+-------------------------------------------------+
| Field             | Value                                           |
+-------------------+-------------------------------------------------+
| allocation_pools  | {"start": "10.9.0.100", "end": "10.9.255.254"} |
| cidr              | 10.9.0.0/16                                     |
| dns_nameservers   | 141.85.241.15                                   |
| enable_dhcp       | True                                            |
| gateway_ip        | 10.9.0.1                                        |
| host_routes       |                                                 |
| id                | 3f7ca0ff-7855-4f12-b6b4-4c4a763aa22f            |
| ip_version        | 4                                               |
| ipv6_address_mode |                                                 |
| ipv6_ra_mode      |                                                 |
| name              | 10_9                                            |
| network_id        | 424666ed-c0e8-4d1c-96fe-c22c56262a87            |
| subnetpool_id     |                                                 |
| tenant_id         | 975459f77464498898a1b17c8f08c8d4                |
```

```
+-------------------+-----------------------------------------------------+
```

Security groups

Security groups are also handled by **Neutron**, so we'll use **neutron security-group-list**.

```
$ neutron security-group-list

+--------------------------------------+---------+-------------------------
----------------------------------------+

| id                                   | name    | security_group_rules
|

+--------------------------------------+---------+-------------------------
----------------------------------------+

|   8ef0e4b7-4543-48da-b304-00f74c6e20c4   |   default   |   egress,   IPv4
|

|                                                  |           | egress, IPv6
|

|                                              |          | ingress, IPv4, 10000-
40000/tcp, remote_ip_prefix: 0.0.0.0/0          |

|                                            |          | ingress, IPv4, 22/tcp,
remote_ip_prefix: 0.0.0.0/0                   |

|                                           |          | ingress, IPv4, 3389/tcp,
remote_ip_prefix: 0.0.0.0/0                |

|                                           |          | ingress, IPv4, 4000/tcp,
remote_ip_prefix: 0.0.0.0/0                |

|                                            |          | ingress, IPv4, 443/tcp,
remote_ip_prefix: 0.0.0.0/0                   |

|                                           |          | ingress, IPv4, 5901/tcp,
remote_ip_prefix: 0.0.0.0/0                |

|                                            |          | ingress, IPv4, 80/tcp,
remote_ip_prefix: 0.0.0.0/0                   |

|                                           |          | ingress, IPv4, 8080/tcp,
remote_ip_prefix: 0.0.0.0/0                |

|                                     |        | ingress, IPv4, remote_group_id:
8ef0e4b7-4543-48da-b304-00f74c6e20c4 |

|                                     |        | ingress, IPv6, remote_group_id:
8ef0e4b7-4543-48da-b304-00f74c6e20c4 |
```

```
+--------------------------------------+---------+----------------------------
----------------------------------------+
```

For very verbose details, use **neutron security-group-show** and the ID of the security group:

```
$ neutron security-group-show 8ef0e4b7-4543-48da-b304-00f74c6e20c4

+----------------------+-------------------------------------------------------
---------------+

| Field              | Value                                                    |

+----------------------+-------------------------------------------------------
---------------+

|   description                                |   Default   security   group
|

|  id                                    |  8ef0e4b7-4543-48da-b304-00f74c6e20c4
|

| name               | default                                                  |

| security_group_rules | {                                                      |

|                  |     "remote_group_id": null,                               |

|                  |     "direction": "ingress",                                |

|                                    |         "remote_ip_prefix":  "0.0.0.0/0",
|

|                  |     "protocol": "tcp",                                     |

|                      |      "tenant_id": "e81c0aa57f61461c8da8496157f7041e",
|

|                  |     "port_range_max": 4000,                                |

|                       |      "security_group_id": "8ef0e4b7-4543-48da-b304-
00f74c6e20c4",  |

|                  |     "port_range_min": 4000,                                |

|                  |     "ethertype": "IPv4",                                   |

|                         |       "id": "013de936-4790-477c-98bf-631ae252e60a"
|

|                   | }                                                        |

[...]
```

Booting the instance

Finally, after listing all the parameters, we can boot the instance. We will use:

- **image**: **Ubuntu 16.04 Xenial** (with ID **3672370a-af54-47c2-b2c1-d9875952415f**)
- **flavor**: **m1.tiny** (with ID **3c183fea-cea6-489e-8b6e-d34c4bf073ec**)
- **keypair**: your own keypair
- **network**: **vlan9** (with ID **424666ed-c0e8-4d1c-96fe-c22c56262a87**)
- **security-group**: **default**
- **name**: **scgc**

For booting, we use **nova boot**:

```
$ nova boot --flavor m1.tiny --image 3672370a-af54-47c2-b2c1-d9875952415f \

--nic net-id=424666ed-c0e8-4d1c-96fe-c22c56262a87 --security-group default --
key-name fep scgc

+--------------------------------------+--------------------------------------
----------------------+

|  Property                                                                 |  Value
|

+--------------------------------------+--------------------------------------
----------------------+

|  OS-DCF:diskConfig                                                      |  MANUAL
|

|    OS-EXT-AZ:availability_zone                                                |
|

|   OS-EXT-SRV-ATTR:host                                                    |   -
|

|          OS-EXT-SRV-ATTR:hypervisor_hostname                        |          -
|

|   OS-EXT-SRV-ATTR:instance_name                               |   instance-00007e32
|

|   OS-EXT-STS:power_state                                                  |   0
|

|  OS-EXT-STS:task_state                                                 |  scheduling
|
```

```
|   OS-EXT-STS:vm_state                                                  |   building
|
|    OS-SRV-USG:launched_at                                                |    -
|
|    OS-SRV-USG:terminated_at                                              |    -
|
|   accessIPv4                                                                |
|
|   accessIPv6                                                                |
|
|  adminPass                                                            |  MSM9keuqK8zB
|
|   config_drive                                                              |
|
|  created                                                      |  2018-05-14T20:37:28Z
|
| flavor                                    | m1.tiny (3c183fea-cea6-489e-8b6e-
d34c4bf073ec)             |
|   hostId                                                                    |
|
| id                                       | acd0fcdd-ac58-49b5-ad04-bf34cc7af4a7
|
| image                                     | Ubuntu 16.04 Xenial (3672370a-af54-
47c2-b2c1-d9875952415f) |
|   key_name                                                                |  fep
|
|   metadata                                                                 |  {}
|
|   name                                                                   |  scgc
|
|             os-extended-volumes:volumes_attached             |             []
|
|   progress                                                                 |  0
|
|   security_groups                                                      |  default
|
|   status                                                                 |  BUILD
|
```

```
| tenant_id                                | e81c0aa57f61461c8da8496157f7041e
|

| updated                                                | 2018-05-14T20:37:29Z
|

| user_id                                                  | alexandru.carp
|

+------------------------------------+------------------------------------
----------------------+
```

In **Horizon**, follow the state of the booted instance.
Note the ID of the instance, because we'll use it later.

Instance lifecycle

In this section, we will perform various operations regarding the lifecycle of an instance.

Query

We can use **nova list** for listing all instances:

```
$ nova list

+--------------------------------------+------+--------+------------+--------
-----+---------------------+

| ID                                   | Name | Status | Task State | Power
State | Networks            |

+--------------------------------------+------+--------+------------+--------
-----+---------------------+

| acd0fcdd-ac58-49b5-ad04-bf34cc7af4a7 | scgc | ACTIVE | -          | Running
| vlan9=10.9.119.119 |

+--------------------------------------+------+--------+------------+--------
-----+---------------------+
```

With **nova show**, we can get details:

```
$ nova show acd0fcdd-ac58-49b5-ad04-bf34cc7af4a7

+------------------------------------+------------------------------------
----------------------+

|  Property                                                          |  Value
|

+------------------------------------+------------------------------------
----------------------+
```

```
|   OS-DCF:diskConfig                                                  |   MANUAL
|
|    OS-EXT-AZ:availability_zone                                       |    GP
|
|  OS-EXT-SRV-ATTR:host                                |  quad-wn20.grid.pub.ro
|
|     OS-EXT-SRV-ATTR:hypervisor_hostname          |     quad-wn20.grid.pub.ro
|
|   OS-EXT-SRV-ATTR:instance_name                          |   instance-00007e32
|
|   OS-EXT-STS:power_state                                             |   1
|
|   OS-EXT-STS:task_state                                              |   -
|
|   OS-EXT-STS:vm_state                                                |   active
|
|  OS-SRV-USG:launched_at                          |  2018-05-14T20:37:54.000000
|
|    OS-SRV-USG:terminated_at                                          |    -
|
|   accessIPv4                                                                |
|
|   accessIPv6                                                                |
|
|   config_drive                                                              |
|
|  created                                                 |  2018-05-14T20:37:28Z
|
| flavor                                     | m1.tiny (3c183fea-cea6-489e-8b6e-
d34c4bf073ec)             |
|  hostId                                                                     |
e1774fd65778cdf1c7aaeb0240bdf46d071197d067358e6fea3e09e8   |
| id                                       | acd0fcdd-ac58-49b5-ad04-bf34cc7af4a7
|
| image                                     | Ubuntu 16.04 Xenial (3672370a-af54-
47c2-b2c1-d9875952415f) |
|   key_name                                                           |   fep
|
```

```
|  metadata                                                                  |  {}
|
|  name                                                                      |  scgc
|
|             os-extended-volumes:volumes_attached             |             []
|
|  progress                                                                  |  0
|
|  security_groups                                                           |  default
|
|  status                                                                    |  ACTIVE
|
| tenant_id                                  | e81c0aa57f61461c8da8496157f7041e
|
|  updated                                                   |  2018-05-14T20:37:54Z
|
|  user_id                                                   |  alexandru.carp
|
|  vlan9  network                                            |  10.9.119.119
|
+------------------------------------+-------------------------------------
----------------------+
```

Test connectivity to the instance using **ssh** (user **ubuntu**).

```
$ ssh -i /path/to/key ubuntu@<INSTANCE IP>
```

Stop

For stopping the instance, without deleting it, we can use the **nova stop** command. This is the equivalent of shutting down the instance.

```
$ nova stop <INSTANCE ID>

Request to stop server acd0fcdd-ac58-49b5-ad04-bf34cc7af4a7 has been accepted.

$ nova list

+--------------------------------------+------+---------+------------+-------
------+--------------------+

| ID                                   | Name | Status  | Task State | Power
State | Networks           |
```

```
+--------------------------------------+------+---------+------------+-------
------+--------------------+

| acd0fcdd-ac58-49b5-ad04-bf34cc7af4a7 | scgc | SHUTOFF | -          | Shutdown
| vlan9=10.9.119.119 |

+--------------------------------------+------+---------+------------+-------
------+--------------------+
```

- Also, check in Horizon that the instance has been stopped.
- With **ssh**, check that the instance is no longer reachable.

Start

After being stopped, an instance can be started with the **nova start** command:

```
$ nova start <INSTANCE ID>

Request to start server acd0fcdd-ac58-49b5-ad04-bf34cc7af4a7 has been accepted.

$ nova list

+--------------------------------------+------+--------+------------+--------
-----+--------------------+

| ID                                   | Name | Status | Task State | Power
State | Networks           |

+--------------------------------------+------+--------+------------+--------
-----+--------------------+

| acd0fcdd-ac58-49b5-ad04-bf34cc7af4a7 | scgc | ACTIVE | -          | Running
| vlan9=10.9.119.119 |

+--------------------------------------+------+--------+------------+--------
-----+--------------------+
```

After starting the instance:

- Check in Horizon that the instance has been started.
- With **ssh**, check that the instance is reachable.

Terminate

Terminate the instance with the **nova delete** command:

```
$ nova delete <INSTANCE ID>

Request to delete server acd0fcdd-ac58-49b5-ad04-bf34cc7af4a7 has been
accepted.
```

After that, the instance should not appear in **nova list** any more:

```
$ nova list
```

```
+----+------+--------+------------+-------------+----------+

| ID | Name | Status | Task State | Power State | Networks |

+----+------+--------+------------+-------------+----------+

+----+------+--------+------------+-------------+----------+
```

Initial configuration

OpenStack also provides a mechanism for configuring instances at first boot. This is also called **day0 configuration** and is implemented by the **cloud-init** module.
First, you must create a file that is actually a script (on fep) that will be injected in the instance and run at boot time:

```
$ cat day0.txt

#!/bin/bash

echo test > /tmp/test.txt
```

Then, boot a instance with **nova boot**, but specifying the **user-data** parameter:

```
$ nova boot --flavor m1.tiny --image 3672370a-af54-47c2-b2c1-d9875952415f --
nic net-id=424666ed-c0e8-4d1c-96fe-c22c56262a87 --security-group default --
key-name fep --user-data ./day0.txt myinstance
```

After the instance finished booting, login as ubuntu as verify that the **/tmp/test.txt** has been created.
Note: Any script, no matter how complex, can be injected into the instance using this mechanism.
Terminate the instance after moving to the next task.

Networking

We want to create a topology of 2 VMs (a **client** and a **server**), connected through a private network. Each VM should also have a management connection to the **vlan9** network:

```
+--------+        mynetwork        +--------+

| client |-------------------------| server |

+--------+       172.16.1.0/24     +--------+

    |                                   |

    |                                   |

    | vlan9                             | vlan9
```

- **vlan9** already exists and is a **provider** (physical) network

- **mynetwork** will have to be created and will be a **self-service** network (user defined, only visible inside our own tenant)

Creating the network

Create the network using the **neutron net-create** command:

```
$ neutron net-create mynetwork

Created a new network:

+-----------------+--------------------------------------+

| Field           | Value                                |

+-----------------+--------------------------------------+

| admin_state_up  | True                                 |

| id              | 20a1cce9-9adc-48d3-bb55-3917dd3fbdea |

| mtu             | 0                                    |

| name            | mynetwork                            |

| router:external | False                                |

| shared          | False                                |

| status          | ACTIVE                               |

| subnets         |                                      |

| tenant_id       | e81c0aa57f61461c8da8496157f7041e     |

+-----------------+--------------------------------------+
```

Verify it was successfully created using:

- Horizon, in **Project** → **Network** → **Networks**
- the **neutron net-show** command:

```
$ neutron net-show <NETWORK ID>

+-----------------+--------------------------------------+

| Field           | Value                                |

+-----------------+--------------------------------------+

| admin_state_up  | True                                 |

| id              | 20a1cce9-9adc-48d3-bb55-3917dd3fbdea |
```

```
| mtu              | 0                                    |
| name             | mynetwork                            |
| router:external  | False                                |
| shared           | False                                |
| status           | ACTIVE                               |
| subnets          |                                      |
| tenant_id        | e81c0aa57f61461c8da8496157f7041e     |
+------------------+--------------------------------------+
```

Creating the subnet

The next step is to create a subnet for **mynetwork**. We will use **neutron subnet-create** and:

- **172.16.1.0/24** for prefix
- **mysubnet** for name
- no gateway (VMs will have the gateway through **vlan9**)

```
$ neutron subnet-create --name mysubnet --no-gateway mynetwork 172.16.1.0/24
Created a new subnet:
+-------------------+------------------------------------------------+
| Field             | Value                                          |
+-------------------+------------------------------------------------+
| allocation_pools  | {"start": "172.16.1.1", "end": "172.16.1.254"} |
| cidr              | 172.16.1.0/24                                  |
| dns_nameservers   |                                                |
| enable_dhcp       | True                                           |
| gateway_ip        |                                                |
| host_routes       |                                                |
| id                | b5b278b8-5d80-4285-9efd-094c6481b6e1           |
| ip_version        | 4                                              |
| ipv6_address_mode |                                                |
| ipv6_ra_mode      |                                                |
```

```
| name              | mysubnet                                          |
| network_id        | 20a1cce9-9adc-48d3-bb55-3917dd3fbdea              |
| subnetpool_id     |                                                   |
| tenant_id         | e81c0aa57f61461c8da8496157f7041e                  |
+-------------------+---------------------------------------------------+
```

Verify the subnet was successfully created using Horizon, in **Project** → **Network** → **Networks** → **mynetwork** → **Subnets**

Booting the instances

We will boot the instances according to the topology. Note that each instance will have 2 vNICs:

- the first one in **vlan9**
- the second one in **mynetwork**

```
$ nova boot --flavor m1.tiny --image 3672370a-af54-47c2-b2c1-d9875952415f --nic net-id=424666ed-c0e8-4d1c-96fe-c22c56262a87 --nic net-id=<mynetwork ID> --security-group default --key-name <keypair> client



$ nova boot --flavor m1.tiny --image 3672370a-af54-47c2-b2c1-d9875952415f --nic net-id=424666ed-c0e8-4d1c-96fe-c22c56262a87 --nic net-id=<mynetwork ID> --security-group default --key-name <keypair> server
```

Testing connectivity

We need to login via SSH on each instance and trigger the DHCP client for the second NIC:

```
$ sudo dhclient ens4
```

Verify that each instance gets the correct IP address and verify connectivity between each other using **ping**.

Automatic configuration

Terminate the instances and launch them again, but this time instead on logging in to the instances and manually triggering dhclient, do this via **cloud-init**.

Releasing resources

Delete the resources in the reverse order:

- terminate the instances:

```
$ nova delete <client instance ID>
```

```
$ nova delete <server instance ID>
```

- delete the network:

```
$ neutron net-delete <mynetwork ID>
```

Verify that the resources were deleted using **nova list** and **neutron net-list**.

Orchestration

Using orchestration, we can create multiple cloud objects through a single operation. For this, we need an additional object, called **stack**. The service that handles orchestration in OpenStack is **Heat**.
We will define a new stack that deploys 3 Ubuntu VMs at the same time. For this, go to **Project** → **Orchestration** → **Stacks** and click on **Launch Stack**
For **Template source**, upload a file with the following content (substitute <KEYPAIR NAME> with your own keypair name):

```
heat_template_version: 2013-05-23


resources:
  vm1:
    type: OS::Nova::Server
    properties:
      name: vm1
      image: 3672370a-af54-47c2-b2c1-d9875952415f
      flavor: m1.tiny
      key_name: <KEYPAIR NAME>
      networks:
      - network: vlan9


  vm2:
    type: OS::Nova::Server
    properties:
      name: vm2
      image: 3672370a-af54-47c2-b2c1-d9875952415f
```

```
      flavor: m1.tiny
      key_name: <KEYPAIR NAME>
      networks:
      - network: vlan9

  vm3:
    type: OS::Nova::Server
    properties:
      name: vm3
      image: 3672370a-af54-47c2-b2c1-d9875952415f
      flavor: m1.tiny
      key_name: <KEYPAIR NAME>
      networks:
      - network: vlan9
```

Stack operations

After the stack is created:

- Verify that 3 instances have been launched in **Nova**.
- Click on the stack name and inspect the associated resources.
- **Suspend** / **Resume** the stack and see what happens with the instances.
- Delete the stack.

Initial configuration

Starting from the above template, create a new one that also provisions the instances with an initial configuration:

- Each instance should have **apache2** installed.
- The **index.html** file in **/var/www/html** should contain **This is <VM name>**.

Hint: https://docs.openstack.org/heat/queens/template_guide/software_deployment.html

Security in private clouds: Openstack security

In this chapter, we will present a deployment for a **minimal** OpenStack cloud, comprising the basic services. After installation, we will configure security by creating **users**, **tenants** and **roles**.

We will install the following services:

- **Keystone** (identity service)
- **Glance** (image service)
- **Nova** (compute service)

Everything will be installed in an **all-in-one VM**. Also, this VM will serve as a compute node (hosting the hypervisor on which instances will run).
The **all-in-one** approach is only intended for demo and learning setups. It is NOT suitable for a production environment, due to severe performance limitations.

Creating and preparing the VM

In the faculty's OpenStack cloud, launch an instance with the following parameters:

- *Name*: **newton**
- *Availability Zone*: **any**
- *Instance boot source*: **Boot from image**
- *Image name*: **Ubuntu 16.04 Xenial**
- *Flavor*: **m1.large**
- *Keypair*: your own keypair from `fep.grid.pub.ro`

Connect to the VM from `fep.grid.pub.ro`, using the username `ubuntu`.
In `/etc/hosts`, map the IP address of the instance to its hostname:

```
$ cat /etc/hosts
127.0.0.1    localhost
<IP ADDRESS> newton
[...]
```

Repositories

First, do a package upgrade:

```
$ sudo apt update
$ sudo apt upgrade
```

Add the repository with the Ubuntu cloud packages for OpenStack version **Newton**. After that, upgrade the packages again:

```
$ sudo apt install software-properties-common
$ sudo add-apt-repository cloud-archive:newton
$ sudo apt update
$ sudo apt dist-upgrade
```

DO NOT forget to enter the second `apt-get update` command! If you do not enter it, an incorrect version of OpenStack will be installed!

Install the OpenStack client packages and reboot:

```
$ sudo apt install python-openstackclient
```

Reboot:

```
$ sudo reboot
```

Service prerequisites

For OpenStack to function, some additional services are required.

a. RabbitMQ

RabbitMQ is a message queue service that implements the AMQP protocol. It is used by all OpenStack services for asynchronous communication.
Install the package:

```
$ sudo apt install rabbitmq-server
```

Create the credentials for connecting to the service (e.g. username **openstack**, password **student**):

```
$ sudo rabbitmqctl add_user openstack student
```

Grant all permissions to the created user:

```
$ sudo rabbitmqctl set_permissions openstack ".*" ".*" ".*"
```

b. Memcached

Memcached is a memory caching service, used by OpenStack for caching authentication tokens.
Install the packages:

```
$ sudo apt install memcached python-memcache
```

Edit the configuration file (`/etc/memcached.conf`) and modify the line with `-l 127.0.0.1` so that the service will listen on all interfaces:

```
-l 0.0.0.0
```

Restart the service:

```
$ sudo service memcached restart
```

c. SQL Database

All OpenStack services store configuration and management data in an SQL database. For production environments, **MySQL** or **PostgreSQL** are recommended. For conserving the resources, we will use **SQLite**, for which no installation is necessary.
Do not use **SQLite** in a production environment, because it will not scale and performance will be impacted.

Keystone

a. Package

Install the package:

```
$ sudo apt install keystone
```

b. Configuration file

Edit the `/etc/keystone/keystone.conf` in the `[token]` section, configuring the Fernet token provider:

```
[token]
...
provider = fernet
```

c. Database

Populate the Keystone database:

```
$ sudo su -s /bin/sh -c "keystone-manage db_sync" keystone
```

d. Bootstrapping the config

Initialize Fernet key repositories:

```
$ sudo keystone-manage fernet_setup --keystone-user keystone --keystone-group keystone
$ sudo keystone-manage credential_setup --keystone-user keystone --keystone-group keystone
```

Bootstrap the Identity service:

```
$ sudo keystone-manage bootstrap --bootstrap-password admin \
  --bootstrap-admin-url http://newton:35357/v3/ \
  --bootstrap-internal-url http://newton:5000/v3/ \
  --bootstrap-public-url http://newton:5000/v3/ \
  --bootstrap-region-id RegionOne
```

Each OpenStack service has 3 endpoints:

- **admin** endpoint - used for full-access
- **internal** endpoint - on which other services connect
- **public** endpoint - on which normal users connect

Restart the Apache2 service (Keystone runs inside Apache2):

```
$ sudo service apache2 restart
```

e. Create the OpenStack RC file

This OpenStack RC file will be used for authenticating as admin:

```
$ cat admin-openrc
export OS_PROJECT_DOMAIN_NAME=Default
export OS_USER_DOMAIN_NAME=Default
export OS_PROJECT_NAME=admin
export OS_USERNAME=admin
export OS_PASSWORD=admin
export OS_AUTH_URL=http://newton:35357/v3
export OS_IDENTITY_API_VERSION=3
export OS_IMAGE_API_VERSION=2
```

f. The service project

Source the RC file and create the service project:

```
$ source admin-openrc
$ openstack project create --domain default --description "Service Project"
service
+-------------+----------------------------------+
| Field       | Value                            |
+-------------+----------------------------------+
| description | Service Project                  |
| domain_id   | default                          |
| enabled     | True                             |
| id          | b30d814c79cc4991960cf5cca17fafcb |
| is_domain   | False                            |
| name        | service                          |
| parent_id   | default                          |
+-------------+----------------------------------+
```

The **service** project in an internal tenant, in which all OpenStack services take part.

Glance

a. User and role

Create the OpenStack internal `glance` user, under which the `glance` service will run. Choose a password (e.g. `glance`):

```
$ openstack user create --domain default --password-prompt glance
User Password:
Repeat User Password:
+---------------------+----------------------------------+
| Field               | Value                            |
+---------------------+----------------------------------+
| domain_id           | default                          |
| enabled             | True                             |
| id                  | 7086eaefacf94b9fa1e2ee9d329a73c3 |
| name                | glance                           |
```

```
| options              | {}                                |
| password_expires_at | None                               |
+---------------------+-----------------------------------+
```

Grant this user the `admin` role:

```
$ openstack role add --project service --user glance admin
```

This user will just be used internally. External users will not be able to login as `glance`.

b. Service and endpoints

Register the service in the catalog:

```
$ openstack service create --name glance --description "OpenStack Image" image
+-------------+----------------------------------+
| Field       | Value                            |
+-------------+----------------------------------+
| description | OpenStack Image                  |
| enabled     | True                             |
| id          | 2faa251121e74a31b0e133e3250a8ace |
| name        | glance                           |
| type        | image                            |
+-------------+----------------------------------
```

Create the `admin`, `internal` and `public` endpoints:

```
$ openstack endpoint create --region RegionOne image public http://newton:9292
$ openstack endpoint create --region RegionOne image internal http://newton:9292
$ openstack endpoint create --region RegionOne image admin http://newton:9292
```

c. Package

Install the package:

```
$ sudo apt install glance
```

d. Configuration files

Edit `/etc/glance/glance-api.conf` by adding or editing lines.
Some line must be added, while others may need to be modified or uncommented. Pay attention to the config file! Do not remove lines that are not mentioned!

The database specifies SQLite:

```
[database]
connection = sqlite:////var/lib/glance/glance.db
```

The keystone credentials are correct:

```
[keystone_authtoken]
auth_uri = http://newton:5000
auth_url = http://newton:35357
memcached_servers = newton:11211
```

```
auth_type = password
project_domain_name = Default
user_domain_name = Default
project_name = service
username = glance
password = glance

[paste_deploy]
...
flavor = keystone
```

The backend store is `file`:

```
[glance_store]
stores = file,http
default_store = file
filesystem_store_datadir = /var/lib/glance/images/
```

A file backend is not recommended for production environment. A scalable and reliable one, like **Ceph** should be used.

Edit `/etc/glance/glance-registry.conf` in a similar way:

```
[database]
connection = sqlite:////var/lib/glance/glance.db

[keystone_authtoken]
auth_uri = http://newton:5000
auth_url = http://newton:35357
memcached_servers = newton:11211
auth_type = password
project_domain_name = Default
user_domain_name = Default
project_name = service
username = glance
password = glance

[paste_deploy]
# ...
flavor = keystone
```

e. Database

Populate the Glance database:

```
$ sudo su -s /bin/sh -c "glance-manage db_sync" glance
```

Restart the services:

```
$ sudo service glance-registry restart
$ sudo service glance-api restart
```

f. Onboarding an image

Let's download an image for **CirrOS** (a minimal Linux distribution, which can run with as little as 64MB of RAM):

```
$ wget http://download.cirros-cloud.net/0.3.5/cirros-0.3.5-x86_64-disk.img
```

Upload the image in Glance:

```
$ openstack image create "cirros" --file cirros-0.3.5-x86_64-disk.img --disk-format qcow2 --container-format bare --public
```

Verify it was uploaded correctly:

```
$ openstack image list
```

Nova

a. User and role

Create the OpenStack internal `nova` user, under which the `nova` service will run. Choose a password (e.g. `nova`):

```
$ openstack user create --domain default --password-prompt nova
User Password:
Repeat User Password:
+---------------------+----------------------------------+
| Field               | Value                            |
+---------------------+----------------------------------+
| domain_id           | default                          |
| enabled             | True                             |
| id                  | 7d517982cc19409790397c69e0066326 |
| name                | nova                             |
| options             | {}                               |
| password_expires_at | None                             |
+---------------------+----------------------------------+
```

Grant this user the `admin` role:

```
$ openstack role add --project service --user nova admin
```

This user will just be used internally. External users will not be able to login as `nova`.

b. Service and endpoints

Register the service in the catalog:

```
$ openstack service create --name nova --description "OpenStack Compute" compute
```

Create the `admin`, `internal` and `public` endpoints:

```
$ openstack endpoint create --region RegionOne compute public http://newton:8774/v2.1/%\(tenant_id\)s
$ openstack endpoint create --region RegionOne compute internal http://newton:8774/v2.1/%\(tenant_id\)s
$ openstack endpoint create --region RegionOne compute admin http://newton:8774/v2.1/%\(tenant_id\)s
```

c. Packages

Install the packages:

```
$ sudo apt install nova-api nova-conductor nova-scheduler nova-network
```

Multiple packages are needed for nova, because it is a complex service, with multiple components.

d. Configuration files

Edit `/etc/nova/nova.conf` by adding or editing lines.
Some line must be added, while others may need to be modified or uncommented. Pay attention to the config file! Do not remove lines that are not mentioned!

```
[DEFAULT]
transport_url = rabbit://openstack:student@127.0.0.1
auth_strategy = keystone
use_neutron = False
my_ip = <IP ADDRESS OF YOUR VM>

[database]
connection = sqlite:////var/lib/nova/nova.sqlite

[api_database]
connection = sqlite:////var/lib/nova/nova_api.sqlite

[keystone_authtoken]
auth_uri = http://newton:5000
auth_url = http://newton:35357
memcached_servers = newton:11211
auth_type = password
project_domain_name = Default
user_domain_name = Default
project_name = service
username = nova
password = nova

[glance]
api_servers = http://newton:9292

[oslo_concurrency]
lock_path = /var/lib/nova/tmp
```

Due to a bug, we need to remove the line containing **log-dir**.

e. Networking

Due to resource constraints, we will not install **Neutron** and will just use **Nova-Network**. This is a more lightweight service, but deprecated and with much fewer features. Do not use Nova-Network in a production environment!

f. Database

Populate the Nova database:

```
$ sudo su -s /bin/sh -c "nova-manage api_db sync" nova
$ sudo su -s /bin/sh -c "nova-manage db sync" nova
```

Restart the services:

```
$ sudo service nova-api restart
$ sudo service nova-scheduler restart
$ sudo service nova-conductor restart
$ sudo service nova-network restart
```

5. The compute node

Because we cannot create an additional VM, we will configure the **Nova-Compute** service and the hypervisor on the same VM (thus making a hybrid Controller + Compute node).
DO NOT use this kind of configuration in production. Compute nodes need to be separate, physical machines. This hybrid configuration is only suited for learning or demos.

a. Package

Install the package:

```
$ sudo apt install nova-compute
```

b. Configuration file

Edit `/etc/nova/nova-compute.conf` so that the QEMU hypervisor will be used:

```
[libvirt]
...
virt_type = qemu
```

Restart the service:

```
sudo service nova-compute restart
```

c. Verify

Make sure that the VM is recognized as a compute node:

```
$ openstack compute service list
```

Booting an instance

First. create a flavor with 1 vCPU, 64 MB RAM and 1 GB Disk:

```
$ openstack flavor create --id 0 --vcpus 1 --ram 64 --disk 1 m1.nano
```

Boot an instance, with the `m1.nano` flavor and the `cirros` image. The instance will not be connected to a network:

```
$ nova boot --flavor m1.nano --image cirros --nic none my-vm
```

Use `ps` to verify that the QEMU instance is actually running:

```
$ ps -ef | grep qemu
libvirt+  7741     1  6 21:37 ?        00:01:03 /usr/bin/qemu-system-x86_64 -
name instance-00000006 -S -machine pc-i440fx-xenial,accel=tcg,usb=off -cpu
Haswell-noTSX,+abm,+pdpe1gb,+hypervisor,+rdrand,+f16c,+osxsave,+vmx,+ss,+vme -
m 64 -realtime mlock=off -smp 1,sockets=1,cores=1,threads=1 -uuid 1f8ad333-
8d06-4609-b171-dfba1da6e790       -smbios       type=1,manufacturer=OpenStack
Foundation,product=OpenStack    Nova,version=14.1.0,serial=43736bb5-0fa4-48f4-
86c3-3271c47f8b55,uuid=1f8ad333-8d06-4609-b171-dfba1da6e790,family=Virtual
Machine            -no-user-config            -nodefaults            -chardev
socket,id=charmonitor,path=/var/lib/libvirt/qemu/domain-instance-
00000006/monitor.sock,server,nowait                                    -mon
chardev=charmonitor,id=monitor,mode=control -rtc base=utc -no-shutdown -boot
strict=on    -device    piix3-usb-uhci,id=usb,bus=pci.0,addr=0x1.0x2    -drive
file=/var/lib/nova/instances/1f8ad333-8d06-4609-b171-
dfba1da6e790/disk,format=qcow2,if=none,id=drive-virtio-disk0,cache=none      -
device          virtio-blk-pci,scsi=off,bus=pci.0,addr=0x3,drive=drive-virtio-
disk0,id=virtio-disk0,bootindex=1                                      -chardev
file,id=charserial0,path=/var/lib/nova/instances/1f8ad333-8d06-4609-b171-
dfba1da6e790/console.log -device isa-serial,chardev=charserial0,id=serial0 -
chardev pty,id=charserial1 -device isa-serial,chardev=charserial1,id=serial1 -
device usb-tablet,id=input0 -vnc 127.0.0.1:0 -k en-us -device cirrus-
vga,id=video0,bus=pci.0,addr=0x2            -device            virtio-balloon-
pci,id=balloon0,bus=pci.0,addr=0x4 -msg timestamp=on
```

Run basic lifecycle operation on the VM (e.g. query, stop, start, delete). See Lab 09.

Projects, users and roles

a. Preparing

Before moving on, use **nova list** to make sure you have 1 instance running. If you do not, use **nova boot** to bring it up.

```
$ nova list

+--------------------------------------+-------+--------+------------+-------
------+----------+

| ID                                   | Name  | Status | Task State | Power
State | Networks |
+--------------------------------------+-------+--------+------------+-------
------+----------+
| 1f8ad333-8d06-4609-b171-dfba1da6e790 | my-vm | ACTIVE | -          | Running
|          |
+--------------------------------------+-------+--------+------------+-------
------+----------+
```

b. Projects

List the existing projects, using `openstack project list`. You will see the built-in `admin` project and the internal `service` project.

```
$ openstack project list

+----------------------------------+---------+
```

```
| ID                                | Name    |
+-----------------------------------+---------+
| e766f5933dcd45d2bd022a6ef3e01133 | admin   |
| 206cdfa865ce449aade08f7d54611078 | service |
+-----------------------------------+---------+
```

Let's create new custom project, named `students`:

```
$ openstack project create --domain default --description "SCGC Students"
students
+-------------+----------------------------------+
| Field       | Value                            |
+-------------+----------------------------------+
| description | SCGC Students                    |
| domain_id   | default                          |
| enabled     | True                             |
| id          | de60f57b87f14b0e98efcb762fa414a4 |
| is_domain   | False                            |
| name        | students                         |
| parent_id   | default                          |
+-------------+----------------------------------+
```

Let's also set a quota of maximum 1 instance for this project:

```
$ openstack quota set --instances 1 students
```

c. Users

List the current users, using `openstack user list`. You will see the built-in `admin` user and the internal users `nova` and `glance`.

```
$ openstack user list

+----------------------------------+--------+

| ID                               | Name   |
+----------------------------------+--------+
| ead10d46034242b0a1b698756d4cce25 | admin  |
| c739a4c0f45541f2b598ed51d00640bd | glance |
| 9e877d7ba5d640ae9ca801cb94bd3543 | nova   |
+----------------------------------+--------+
```

Let's create 2 more users, named `student1` and `student2`. The password will be `student`:

```
$ openstack user create --domain default --description "Student One" --password
student student1
+---------------------+----------------------------------+
| Field               | Value                            |
+---------------------+----------------------------------+
| description         | Student One                      |
| domain_id           | default                          |
| enabled             | True                             |
| id                  | db7da55870f34a418a581fa4d04c9e6f |
| name                | student1                         |
| password_expires_at | None                             |
+---------------------+----------------------------------+
```

```
$ openstack user create --domain default --description "Student Two" --password
student student2
+---------------------+----------------------------------+
| Field               | Value                            |
+---------------------+----------------------------------+
| description         | Student Two                      |
| domain_id           | default                          |
| enabled             | True                             |
| id                  | d8e8dccfeb1b48428951743c147a50b5 |
| name                | student2                         |
| password_expires_at | None                             |
+---------------------+----------------------------------+
```

Also, create OpenStack RC files for the 2 users:

```
$ cat student1-openrc
export OS_PROJECT_DOMAIN_NAME=Default
export OS_USER_DOMAIN_NAME=Default
export OS_PROJECT_NAME=students
export OS_USERNAME=student1
export OS_PASSWORD=student
export OS_AUTH_URL=http://newton:5000/v3
export OS_IDENTITY_API_VERSION=3
export OS_IMAGE_API_VERSION=2

$ cat student2-openrc
export OS_PROJECT_DOMAIN_NAME=Default
export OS_USER_DOMAIN_NAME=Default
export OS_PROJECT_NAME=students
export OS_USERNAME=student2
export OS_PASSWORD=student
export OS_AUTH_URL=http://newton:5000/v3
export OS_IDENTITY_API_VERSION=3
export OS_IMAGE_API_VERSION=2
```

d. Roles

List the existing roles using `openstack role list`. You will the built-in roles `admin` and `_member_`:

```
$ openstack role list

+----------------------------------+----------+

| ID                               | Name     |
+----------------------------------+----------+
| 86527e5f7b304bcab4d2ff79abd9dfcb | admin    |
| 9fe2ff9ee4384b1894a90878d3e92bab | _member_ |
+----------------------------------+----------+
```

Let's create another role, called `student`:

```
$ openstack role create student

+-----------+----------------------------------+

| Field     | Value                            |
```

```
+-----------+----------------------------------+
| domain_id | None                             |
| id        | 8bed042dbbb2499e831136fb22533d79 |
| name      | student                          |
+-----------+----------------------------------+
```

e. Assigning users to projects

For assigning a user to a project, we just need to add the same role to the user and the project:

```
$ openstack role add --project students --user student1 student
$ openstack role add --project students --user student2 student
```

Verify the role assignments:

```
$ openstack role assignment list --names
+---------+------------------+-------+------------------+--------+-----------
+
| Role    | User             | Group | Project          | Domain | Inherited |
+---------+------------------+-------+------------------+--------+-----------
+
| admin   | admin@Default    |       | admin@Default    |        | False     |
| admin   | glance@Default   |       | service@Default  |        | False     |
| admin   | nova@Default     |       | service@Default  |        | False     |
| student | student1@Default |       | students@Default |        | False     |
| student | student2@Default |       | students@Default |        | False     |
+---------+------------------+-------+------------------+--------+-----------
+
```

f. Multi-tenancy

Let's see what happens when we do actions as other users.

First, authenticate as `student1` and list the instances:

```
$ source student1-openrc
$ nova list
+----+------+--------+------------+-------------+----------+
| ID | Name | Status | Task State | Power State | Networks |
+----+------+--------+------------+-------------+----------+
+----+------+--------+------------+-------------+----------+
```

You will not be able to see the created instance, because it is in another project. Also, seeing all the instances is not possible, because you are not admin:

```
$ nova list --all
ERROR (Forbidden): Policy doesn't allow
os_compute_api:servers:detail:get_all_tenants to be performed. (HTTP 403)
(Request-ID: req-5a445b40-a7cd-4693-b28d-5c416a14ea41)
```

Boot an instance:

```
$ nova boot --flavor m1.nano --image cirros --nic none student1-vm
$ nova list
```

```
+--------------------------------------+-------------+--------+------------+-
------------+----------+
| ID                                   | Name        | Status | Task State |
Power State | Networks |
+--------------------------------------+-------------+--------+------------+-
------------+----------+
| 7198320d-60a8-4571-815b-7727bb65fd51 | student1-vm | ACTIVE | -          |
Running     |          |
+--------------------------------------+-------------+--------+------------+-
------------+----------+
```

Authenticate as `student2` and list the instances:

```
$ source student2-openrc
$ nova list
+--------------------------------------+-------------+--------+------------+-
------------+----------+
| ID                                   | Name        | Status | Task State |
Power State | Networks |
+--------------------------------------+-------------+--------+------------+-
------------+----------+
| 7198320d-60a8-4571-815b-7727bb65fd51 | student1-vm | ACTIVE | -          |
Running     |          |
+--------------------------------------+-------------+--------+------------+-
------------+----------+
```

Your are able to see the instance, because it is in the same project, even if it was started by another user.

Try to boot a new one:

```
$ nova boot --flavor m1.nano --image cirros --nic none student2-vm
ERROR (Forbidden): Quota exceeded for instances: Requested 1, but already used
1 of 1 instances (HTTP 403) (Request-ID: req-d9072d47-13da-4294-8b42-
1cf8faddbbc7)
```

You are not allowed, because you exceeded the quota.

Delete the instance created by `student1`:

```
$ nova delete 7198320d-60a8-4571-815b-7727bb65fd51
Request to delete server 7198320d-60a8-4571-815b-7727bb65fd51 has been
accepted.
```

Then, create your own instance:

```
$ nova boot --flavor m1.nano --image cirros --nic none student2-vm
$ nova list
+--------------------------------------+-------------+--------+------------+-
------------+----------+
| ID                                   | Name        | Status | Task State |
Power State | Networks |
+--------------------------------------+-------------+--------+------------+-
------------+----------+
| 76d8580f-7c62-42ee-8798-d93716288bb2 | student2-vm | ACTIVE | -          |
Running     |          |
```

```
+--------------------------------------+-------------+--------+------------+-
-----------+----------+
```

Now, login as `admin` and list all the instances:

```
$ source admin-openrc
$ nova list --all
+--------------------------------------+------------+----------------------
----------+--------+------------+-------------+----------+
| ID                                   | Name           | Tenant ID
| Status | Task State | Power State | Networks |
+--------------------------------------+------------+----------------------
----------+--------+------------+-------------+----------+
|    1f8ad333-8d06-4609-b171-dfba1da6e790    |    my-vm                        |
e766f5933dcd45d2bd022a6ef3e01133 | ACTIVE | -          | Running     |          |
|       76d8580f-7c62-42ee-8798-d93716288bb2        |        student2-vm        |
de60f57b87f14b0e98efcb762fa414a4 | ACTIVE | -          | Running     |          |
+--------------------------------------+------------+----------------------
----------+--------+------------+-------------+----------+
```

Delete all the instances:

```
$ nova delete 1f8ad333-8d06-4609-b171-dfba1da6e790
Request to delete server 1f8ad333-8d06-4609-b171-dfba1da6e790 has been
accepted.

$ nova delete 76d8580f-7c62-42ee-8798-d93716288bb2
Request to delete server 76d8580f-7c62-42ee-8798-d93716288bb2 has been
accepted.

$ nova list
+----+------+--------+------------+-------------+----------+
| ID | Name | Status | Task State | Power State | Networks |
+----+------+--------+------------+-------------+----------+
+----+------+--------+------------+-------------+----------+
```

Custom roles

We want to define a special role, called `glanceadmin` that will be able to add and delete images in Glance. First, let's create the role:

```
$ openstack role create glanceadmin

+-----------+----------------------------------+

| Field     | Value                            |
+-----------+----------------------------------+
| domain_id | None                             |
| id        | ba3e153d237542b98504dd2e3c083a61 |
| name      | glanceadmin                      |
+-----------+----------------------------------+
```

Then, edit the Glance policy file `/etc/glance/policy.json` so that only this role will be allowed to add or delete images:

```
...
```

```
    "add_image": "role:glanceadmin",
    "delete_image": "role:glanceadmin",
...
```

Now, try to delete the `cirros` image:

```
$ openstack image delete cirros

Failed to delete image with name or ID 'cirros': 403 Forbidden
You are not authorized to complete delete_image action.
    (HTTP 403)
Failed to delete 1 of 1 images.
```

You are not allowed, even if you are `admin`.
Create a new user, named `glanceguru`:

```
$ openstack user create --domain default --description "Glance Guru" --password
gg glanceguru
+---------------------+----------------------------------+
| Field               | Value                            |
+---------------------+----------------------------------+
| description         | Glance Guru                      |
| domain_id           | default                          |
| enabled             | True                             |
| id                  | 0c572b48b242461285a5a1d74eec4244 |
| name                | glanceguru                       |
| password_expires_at | None                             |
+---------------------+----------------------------------+
```

Assign the `glanceadmin` role to the `glanceguru` user:

```
$ openstack role add --project admin --user glanceguru glanceadmin

$ openstack role assignment list --names
+-------------+--------------------+-------+-----------------+---------+----
-------+
| Role        | User               | Group | Project         | Domain  |
Inherited |
+-------------+--------------------+-------+-----------------+---------+----
-------+
| admin       | admin@Default      |       | admin@Default   |         | False
|
| admin       | glance@Default     |       | service@Default |         | False
|
| admin       | nova@Default       |       | service@Default |         | False
|
| student     | student1@Default   |       | students@Default |         | False
|
| student     | student2@Default   |       | students@Default |         | False
|
| glanceadmin | glanceguru@Default |       | admin@Default   |         | False
|
+-------------+--------------------+-------+-----------------+---------+----
-------+
```

Create the OpenStack RC file:

```
$ cat gg-openrc
```

```
export OS_PROJECT_DOMAIN_NAME=Default
export OS_USER_DOMAIN_NAME=Default
export OS_PROJECT_NAME=admin
export OS_USERNAME=glanceguru
export OS_PASSWORD=gg
export OS_AUTH_URL=http://newton:5000/v3
export OS_IDENTITY_API_VERSION=3
export OS_IMAGE_API_VERSION=2
```

Login as `glanceguru` and try to delete the image:

```
$ source gg-openrc

$ openstack image delete cirros
$ openstack image list
```

Now, try to upload it again:

```
$ openstack image create "cirros" --file cirros-0.3.5-x86_64-disk.img --disk-
format qcow2 --container-format bare --public
403 Forbidden
You are not authorized to complete publicize_image action.
    (HTTP 403)
```

Because you specified the `public` flag, you need to allow one more action in the policy:

```
...

    "publicize_image": "role:glanceadmin",
...
$ openstack image create "cirros" --file cirros-0.3.5-x86_64-disk.img --disk-
format qcow2 --container-format bare --public
+------------------+------------------------------------------------------+
| Field            | Value                                                |
+------------------+------------------------------------------------------+
| checksum         | f8ab98ff5e73ebab884d80c9dc9c7290                     |
| container_format | bare                                                 |
| created_at       | 2018-05-19T22:14:56Z                                 |
| disk_format      | qcow2                                                |
| file             | /v2/images/928e2f5a-3983-45d2-b1c1-1a8960fce7dc/file |
| id               | 928e2f5a-3983-45d2-b1c1-1a8960fce7dc                 |
| min_disk         | 0                                                    |
| min_ram          | 0                                                    |
| name             | cirros                                               |
| owner            | e766f5933dcd45d2bd022a6ef3e01133                     |
| protected        | False                                                |
| schema           | /v2/schemas/image                                    |
| size             | 13267968                                             |
| status           | active                                               |
| tags             |                                                      |
| updated_at       | 2018-05-19T22:14:56Z                                 |
| virtual_size     | None                                                 |
| visibility       | public                                               |
+------------------+------------------------------------------------------+
```

Horizon

Install **Horizon** (the OpenStack dashboard).
Hint: You can follow this tutorial.

Hint: In the above tutorial, replace **controller** with the hostname of your VM (**newton**).

For testing, launch `firefox` from the command-line on `fep.grid.pub.ro`.

ISBN: 978-606-515- 955-6