NETWAYS WEB SERVICES

Cloud Services

Erstellt von:

Erstellt am:

Version:

Inhaltsverzeichnis

Cloud Services

Cloud Services based on OpenStack

Getting Started

Getting Started

Start a project and create the first server

Read our Tutorial on how to create your first OpenStack project and server ***here!***

Getting Started

Openstack CLI

Installation

Just install the official OpenStackClient from PyPi:

```
pip install python-openstackclient python-barbicanclient python-cinderclient python-designateclient python-glanceclient python-heatclient python-neutronclient python-novaclient python-octaviaclient python-magnumclient python-heatclient gnocchiclient
```

Recommended: For an easier login, we recommend installing kubelogin. Follow the official instructions for easy installtion.

Requires: Python >=3.8

Load the OpenStack Environment File

- download the generic nws-id-openstack-rc.sh file for NWS-ID or
- download the direct access template under 'Get Started' in your Openstack project in the Customer Interface

Now you can source the environment file. After entering the password you can send requests to the OpenStack-CLI. If you have access to multiple projects you will be asked to choose one of the available projects before you can continue with the CLI commands.

```
$ source nws-id-openstack-rc.sh
...
$ openstack server list
```

The OpenStack Environment File also has the ability to make project switching easier for you.
Just source the environment file again and you will be asked if you would like to switch the project.

```
$ source nws-id-openstack-rc.sh
Access token is still present. Please choose one of the following options:
1) switch project   2) re-authenticate  3) exit
Enter a number: 1
Selected option: switch project

Please select one of your OpenStack projects.

1) 18-openstack-c4dae  2) 29-openstack-a46f4
Enter a number: 2
Selected project: 29-openstack-a46f4
```

If you would like to only list your available projects, you can just execute the following command:

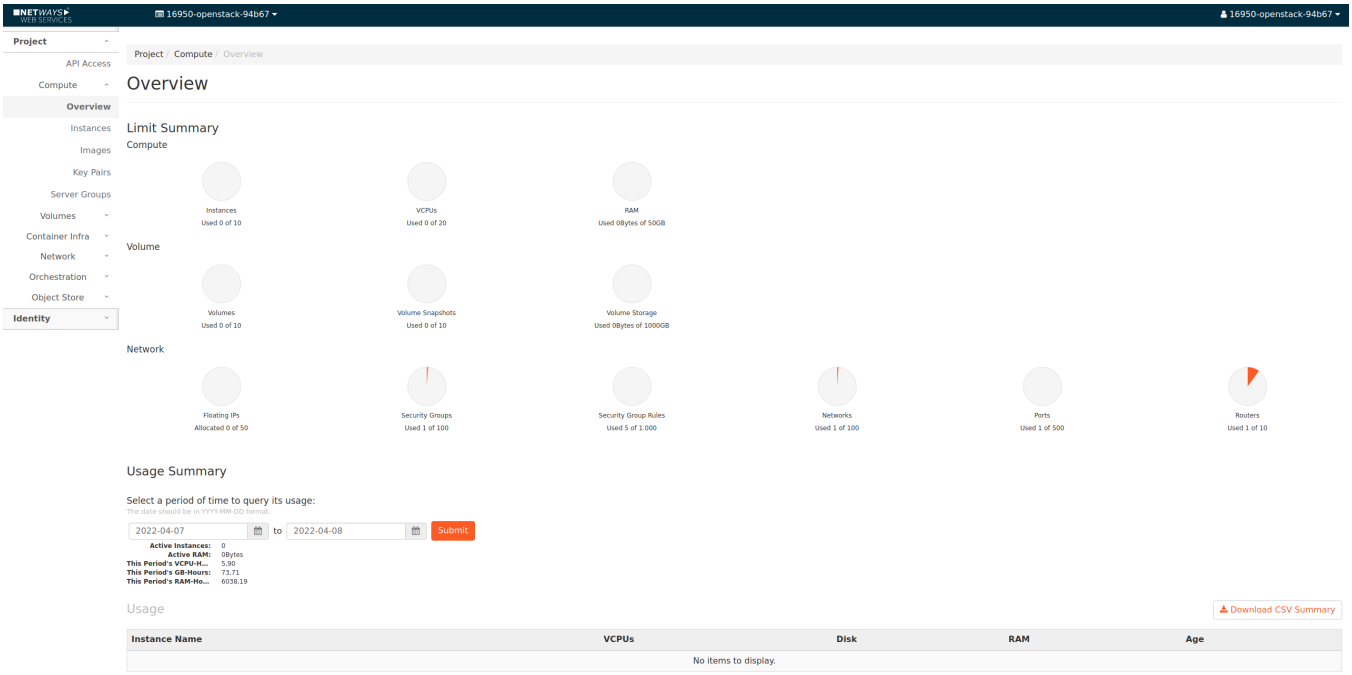
```
$ openstack federation project list
```

Have a look at the Cheat-Sheet to find commonly used CLI commands or have a look into the official OpenStackClient Docs.

Getting Started

Horizon dashboard

The horizon dashboard is the official Web UI for OpenStack. You will find almost anything you need to control your OpenStack cloud there.



You can log in to the horizon dashboard at https://cloud.netways.de with your NWS-ID. Please ask your administrator for the appropriate rights.

Getting Started

S1 and S2 Flavor stacks: Differences between our two VM categories

If you want to start a VM in our OpenStack, you have the choice between two VM categories: S1 and S2.

These differ, apart from hard disk size and compute power (CPU and RAM), mainly in two points: in the IOPS and the bandwidth.

IOPS:

This standard unit of measurement indicates how many input and output commands can be executed per second.
The S1- flavors can process 1000 IOPS in normal operation. For a period of 60 seconds, even a peak of 2000 IOPS can be processed before it is then throttled back to the normal value.
The S2 flavors can process twice the amount of commands - i.e. 2000 IOPS - in normal operation. They can even be turned up to 4000 IOPS for 60 seconds in peak. After that, they are also turned back down to normal.

Bandwidth:

The second difference concerns the network connection. The S1 and S2 categories differ not only in IOPS, but also in the available bandwidth. The larger the bandwidth, the more data can be transferred simultaneously within a time window.
In the S1 flavor, 1 Gbit per second can be transferred as standard. This corresponds to a data volume of 125,000 KB per second. In peak operation, the amount can double to 250,000 KB/s or 2 Gbit/s for a period of 1 minute.
With the S2 Flavor, 2 Gbit/s are available in normal operation. In peak it is even 4 Gbit/s or 500,000 KB/s.

Who uses what?

For all those who have a memory and network-intensive load, it is therefore advisable to fall back on an S2 VM. Examples would be video streams, backup servers (which load a lot of data) or databases. Also AI machines, where large amounts of data need to be processed, should run on s2 flavors.
Here you can find the overview, which helps you to compare all possible flavors in terms of price and conditions.

Administrative Tasks

Administrative Tasks

Backups and Snapshots

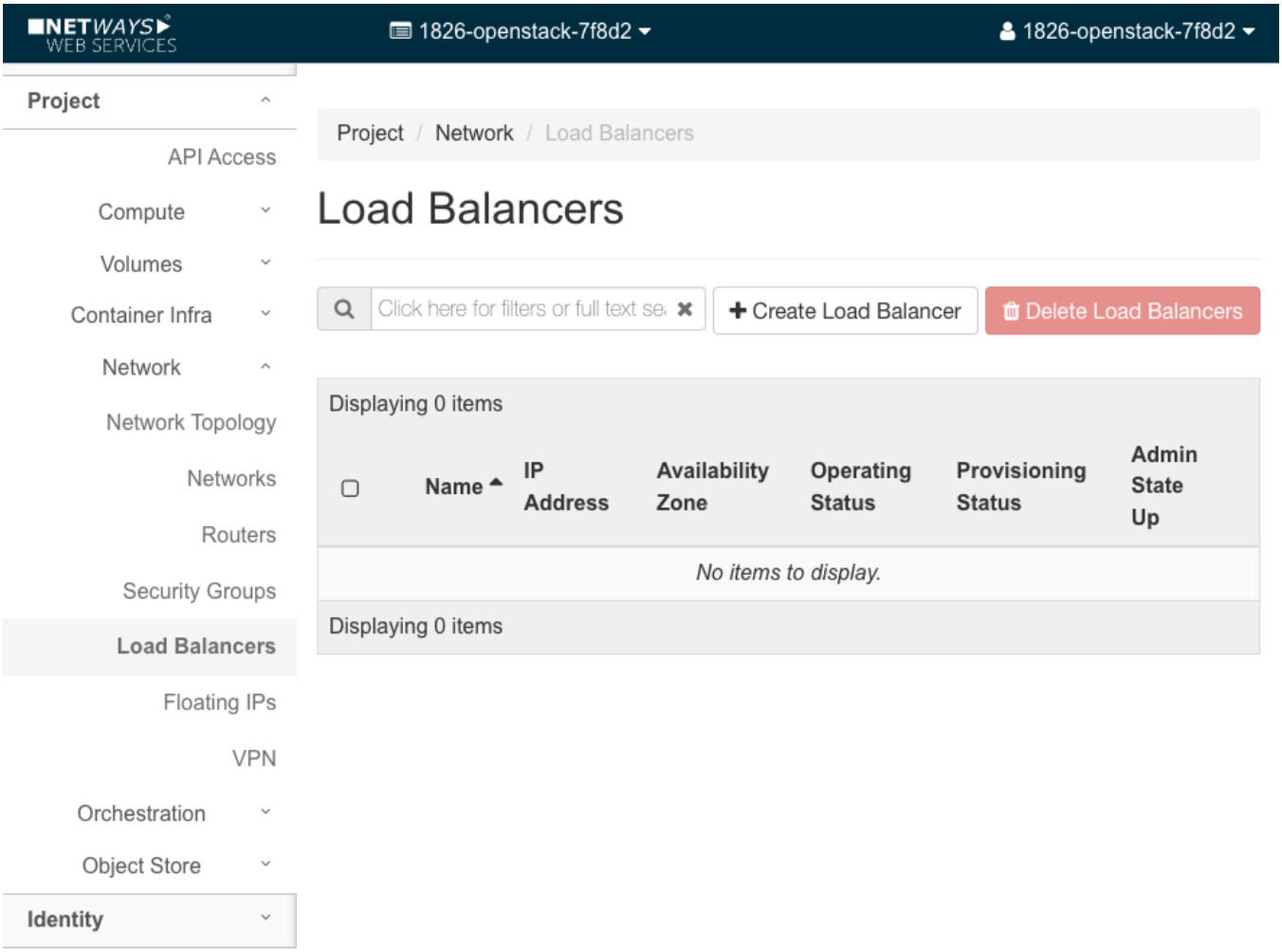
Take a look at the Backup and Snapshot Tutorial to find out how to backup your servers and volumes.

Load Balancers

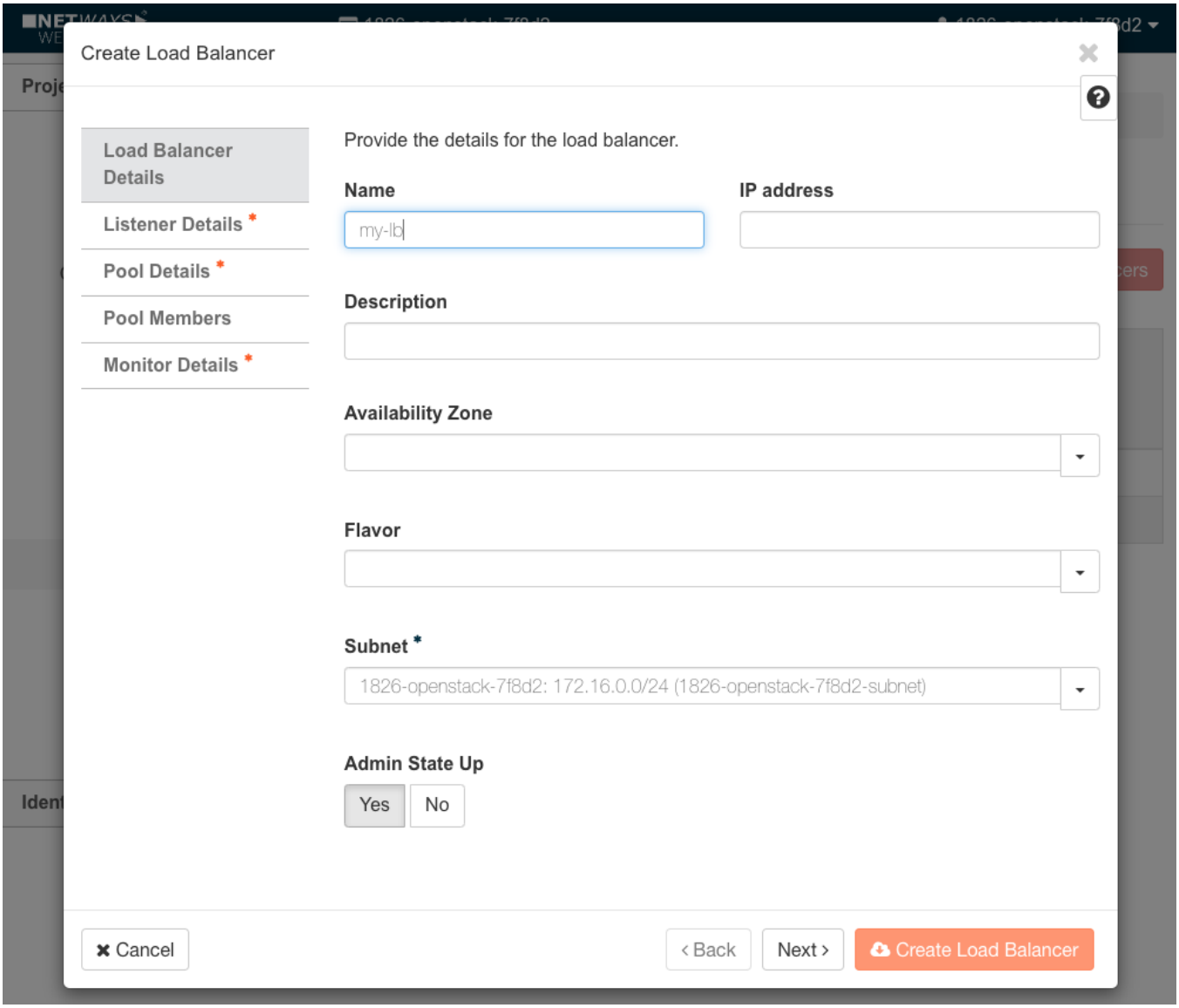
The following guides only cover the basic mandatory settings. If you would like to read more about some advanced load balancer setups and settings, please refer to our Advanced Topics section or the official Basic Load Balancing Cookbook of OpenStack.

Configuration via the OpenStack Dashboard

"Load Balancers" can be found as a subsection of the submenu "Network" in the menu on the left.

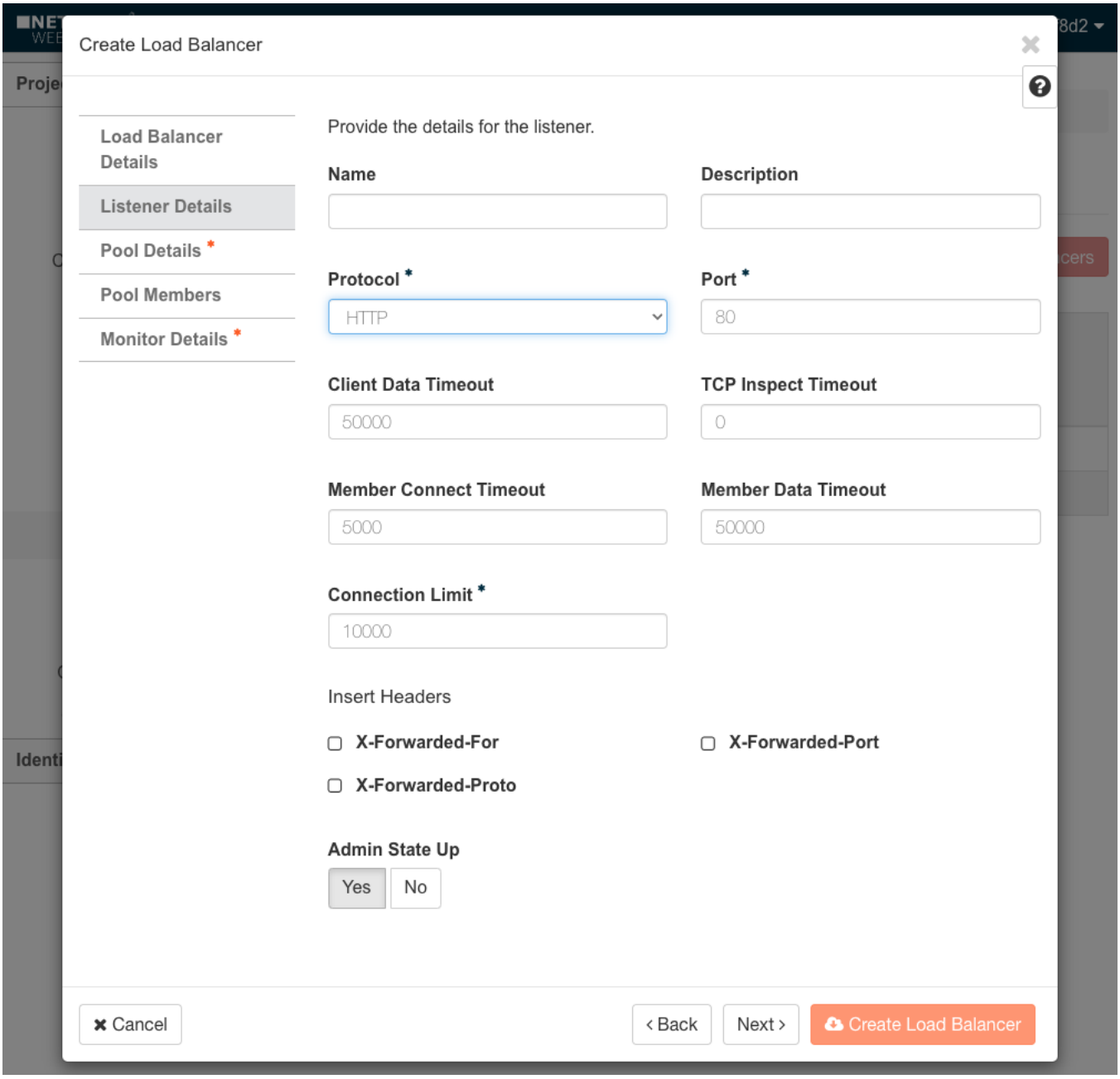


To create a load balancer just click on the button "+ Create Load Balancer". A window will open on that page which will help you to walk through the initial configuration.
On the first page you can fill in some basic details like name and description. It is not necessary but recommended to set a name. However it is mandatory to choose a subnet.
On a plain new OpenStack project you will find your main subnet by looking for a subnet which has the same name as your OpenStack project but with a "-subnet" extension.
To read more about the meaning of the advanced fields, you can click on the question mark on the right side of the window.



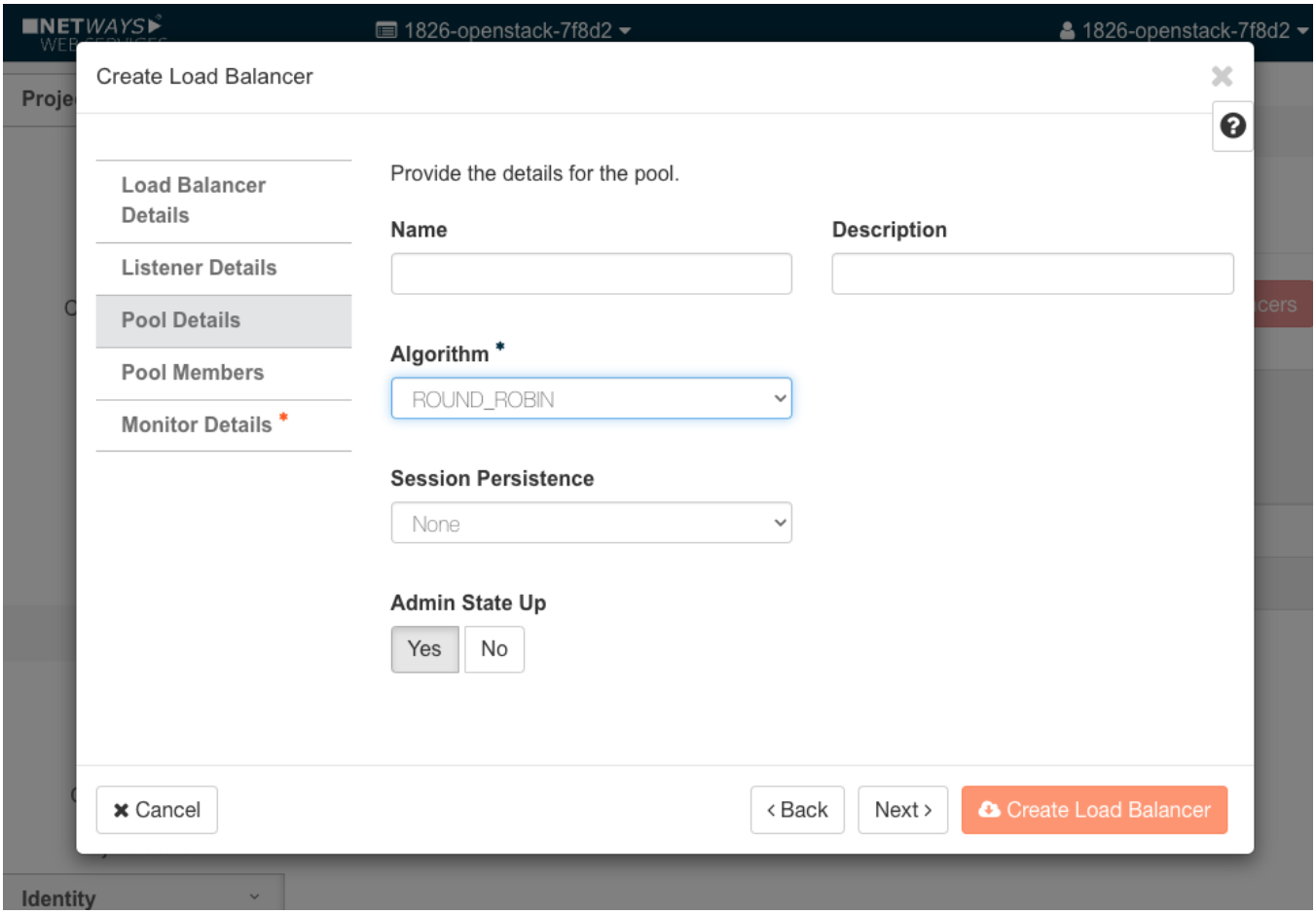
Click on the "Next" button to continue.

On the following page you will be asked to fill in the listener details. You don't have to set a name. But make sure to choose a protocol and to set a port number. The listener defines the frontend listening port of the load balancer. Let's say you would like to deploy a load balancer for some web servers. Then you would choose HTTP or TCP as protocol and set the port to 80 or 443.

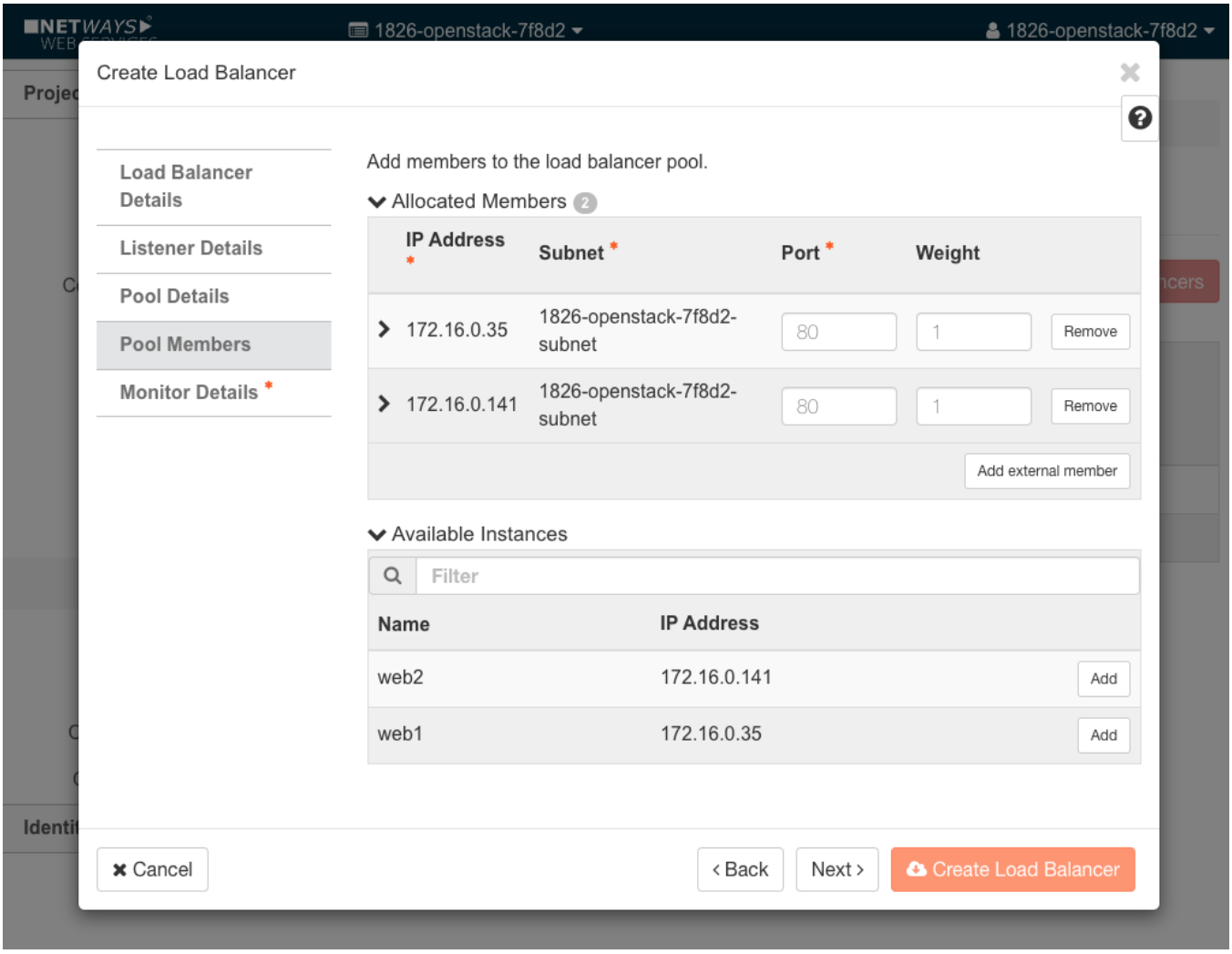


By default the "Connection Limit" is set to "-1" which means, that no limit is enforced. However we discovered that setting no limit here can cause problems.

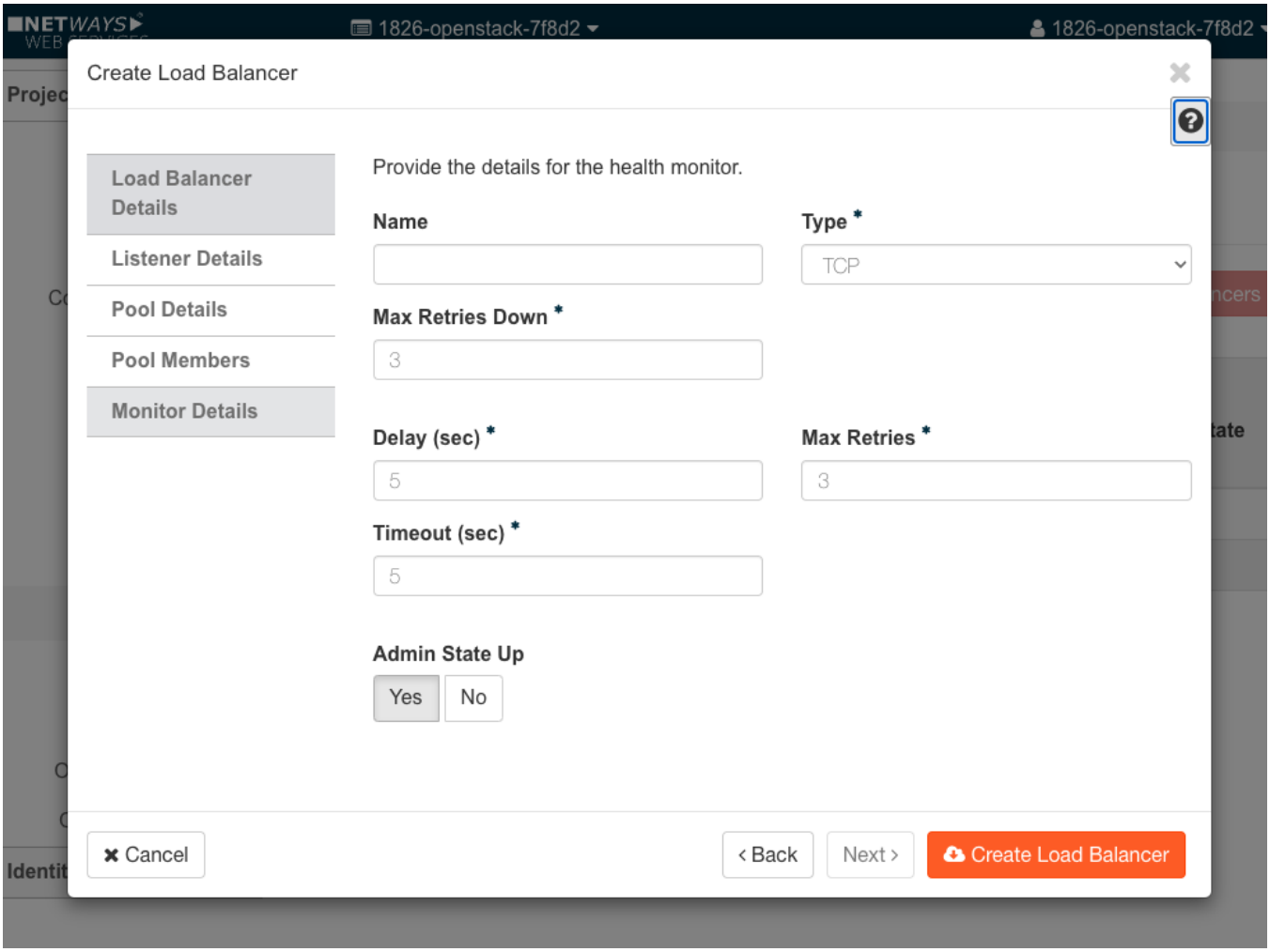
After clicking on "Next", you will be able to choose a load balancing algorithm - for default setups we recommend setting it to "ROUND_ROBIN.



In the following section "Pool Members" you will be asked to assign members to the pool. The members resemble the VMs you would like to forward incoming requests to.
Additionally you will have to set the port number where the application or web server is listening on inside of the corresponding VM.
You can also set a weight - members with a higher weight will receive a higher amount of requests than members with lower weight.

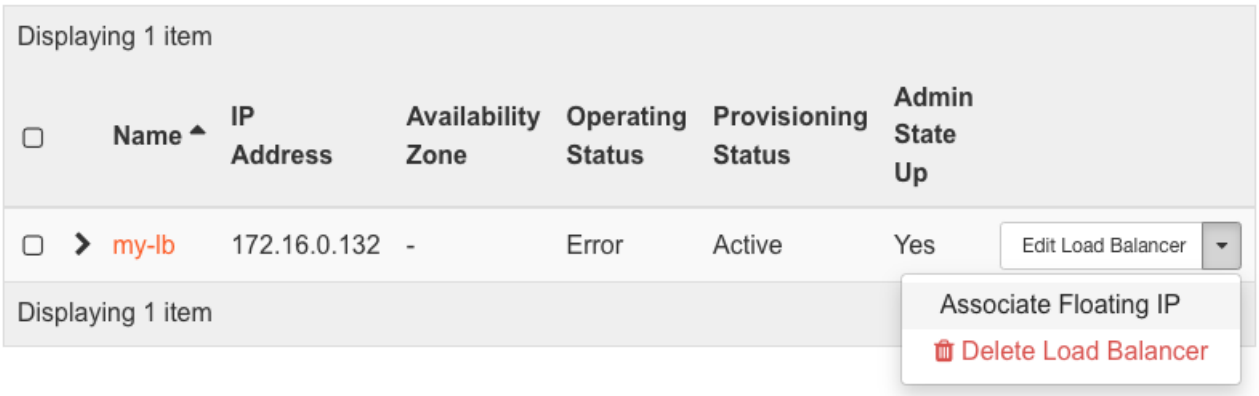


The last section will let you define a health monitor for the pool and its members. Depending on your application or web server make sure to choose an appropriate protocol type. Additionally you can set some timeouts and delay parameter according to which the health monitor will execute the health checks against the pool members.
Reminder: you can click on the question mark on the right side of the window to read more about these parameters.

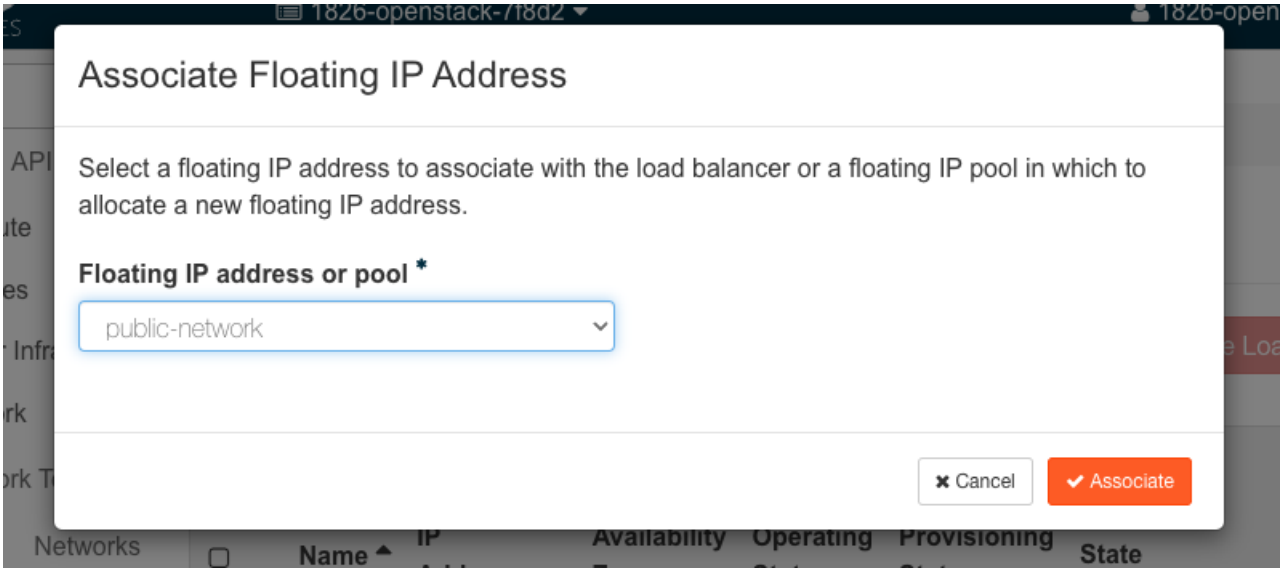


Now that you have filled in all the mandatory fields, you are good to go and therefore you can click on the orange button "Create Load Balancer".
It will take a while until the load balancer and all its dependent resources have been deployed.

As a final optional action you can associate a floating IP with your load balancer. Doing so will let you expose your load balancer to the public internet. Be aware that everyone on the internet would be able to access your application or web site running on your VMs if you assign a floating IP.
To associate a floating IP, click on the dropdown menu of your load balancer and choose "Associate Floating IP".



Depending on if you already have floating IPs allocated to your OpenStack project, you can either choose a existing floating IP or allocate a new one from the window that opens up next.
If you choose to allocate a new floating IP from a pool, make sure to select the public-network.



Finally click on "Associate".
That's it - you have finished the basic configuration of your load balancer.

Configuration via the OpenStack-API using the CLI-Client

To get started with the OpenStack CLI-Client, please refer to our article over here.

The first step after the installation is to get a list of the available subnets in your OpenStack project.
You will need a subnet where you can connect your Load Balancer to. Usually this will be the same subnet where your VMs are connected to, as the Load Balancer will forward the incoming requests from the internet to your VMs.
The output of the subnet list could look similar to the following:

```
~ openstack subnet list
+--------------------------------------+---------------------------+--------------------------------------+--------------------+
| ID                                   | Name                      | Network                              | Subnet             |
+--------------------------------------+---------------------------+--------------------------------------+--------------------+
| 220b5e2f-e086-4c00-8ddb-fbbd539da3ce | public-subnet-4-1         | 210e26d7-942f-4e17-89b2-571eee87d7e4 | 185.11.255.0/24    |
| cd9abbed-fe5b-40a4-b53c-1d42d2732594 | public-subnet-4-2         | 210e26d7-942f-4e17-89b2-571eee87d7e4 | 91.198.2.0/24      |
| ec7b7b20-0bcb-4dc9-87cf-11249689bfd5 | public-subnet-6           | 210e26d7-942f-4e17-89b2-571eee87d7e4 | 2a02:ed80:0:3::/64 |
| f12a110e-b906-4433-85ae-209f5d283bd7 | 1826-openstack-7f8d2-subnet | 3f3089ec-9f19-4082-892a-3fbeaf6a92de | 172.16.0.0/24      |
+--------------------------------------+---------------------------+--------------------------------------+--------------------+
```

As you can see, you should have access to multiple public subnet aswell as one project-specific subnet if you are starting from a new project. The ID of the project-specific subnet is what is relevant for the next step. You can identify your project-specific subnet by looking at the name of the subnet. The default project-specific subnet starts with your project name (here "1826-openstack-7f8d2") and has a "-subnet" extension at the end.
Now copy the ID of the subnet to your clipboard.

Next you can initiate the creation of your Load Balancer. To do so, just execute the following command but replace the argument of "--vip-subnet-id" with your own subnet ID:

```
~ openstack loadbalancer create --name my-lb --vip-subnet-id f12a110e-b906-4433-85ae-209f5d283bd7
```

Before you continue with the configuration of your Load Balancer, you will have to wait until it is up and running.
You can check the status of your Load Balancer to see if it is already operational by executing the following command:

```
~ openstack loadbalancer status show my-lb
```

In the output you should look for "provisioning_status": "ACTIVE" and "operating_status": "ONLINE". As soon as you see those values, you are good to go on with the next steps.

Now that your Load Balancer is up and running, you will need to configure a frontend listening port called "listener". You can choose from a variety of protocol types for the listener (TCP, HTTP, HTTPS, TERMINATED_HTTPS, UDP) and you will also have to specify a port number.
The following example shows how you would create a listener which will listen on incoming HTTP requests on port 80:

```
~ openstack loadbalancer listener create --name listener1 --protocol HTTP --protocol-port 80 my-lb
```

The last parameter needs to be the name of your Load Balancer (here "my-lb").

Continuing with the setup of your Load Balancer, you will have to create a so-called "pool". A pool usually holds multiple "members", which in turn consist of IPs and ports of VMs. In terms of HAProxy configuration, the pool is the equivalent of a backend. The "members" of the pool are therefore the same as the backend servers.
In order to configure a pool you will also have to set a load balancing algorithm which basically is a strategy according to which the incoming requests will be forwarded to the members. A commonly used algorithm is Round Robin. You can choose from the following algorithms: SOURCE_IP, ROUND_ROBIN, LEAST_CONNECTIONS, SOURCE_IP_PORT.
The command to create the pool using ROUND_ROBIN, would look like this:

```
~ openstack loadbalancer pool create --name pool1 --lb-algorithm ROUND_ROBIN --listener listener1 --protocol HTTP
```

A load balancer pool should also be configured with a health monitor. The health monitor regularly checks the reachability of the pool members. In case of one of the members becoming unreachable the requests will only be forwarded to members which are still reachable. But you will have to specify a few parameters to define how the health monitor should act in such circumstances.
The meaning of the parameters is being described after the command.

```
~ openstack loadbalancer healthmonitor create --delay 5 --max-retries 4 --timeout 10 --type HTTP --url-path / pool1
```

The first parameter "--delay" is the seconds to wait between health checks. The parameter "--max-retries" consists of the maximum number of failed health check attempts until the member will be labeled as unreachable. The "--timeout" parameter defines the seconds to wait for a single health check in order to complete. With "--type" you can specify which protocol should be used for the health checks. You can choose from PING, HTTP, TCP, HTTPS, TLS-HELLO, UDP-CONNECT and SCTP. Finally, when choosing "HTTP" or "HTTPS" as type, you can specify a "--url-path" which the name already indicates defines the HTTP path the health check should try to access.

To add members to the pool you will need to supply the subnet ID from the beginning once again on the next command.
Since the member is a representation of a VM you will also have to check for the IP of the VM you would like to forward the incoming requests to. Lastly you will also need to supply the protocol port number - for a web server this would usually be port 80 or port 443.
Here is an example of how the command should look like:

```
~ openstack loadbalancer member create --subnet-id f12a110e-b906-4433-85ae-209f5d283bd7 --address
172.16.0.11 --protocol-port 80 pool1
```

Normally you would have at least two members in a pool. To add more members, just replace the IP address with the one of the other VM and execute the command again.

At last you can ensure that your load balancer is publicly reachable by attaching a floating IP to its virtual port.
This however is an optional step which you can skip if you intend to use the load balancer only for internal connections.

If you have allocated floating IPs before, you can check if you already have unbound floating IPs in your project and reuse one of them. To get a list of all floating IPs allocated in your project, execute the following command:

```
 ~ openstack floating ip list
+--------------------------------------+---------------------+-----------------+--------------------------------------+--------------------------------------
-+----------------------------------+
| ID                                   | Floating IP Address | Fixed IP Address | Port                                 | Floating Network
| Project                              |
+--------------------------------------+---------------------+-----------------+--------------------------------------+--------------------------------------
-+----------------------------------+
| 9fdba43e-5978-4d13-b9b0-6022140bd437 | 91.198.2.199        | 172.16.0.35      | 8ef689d1-a89d-42b2-8af4-
```

```
5a700484ad58 | 210e26d7-942f-4e17-89b2-571eee87d7e4 | f8683d921d4b499eade654f5546c2875 |
+--------------------------------------+---------------------+-----------------+--------------------------------------+--------------------------------------
-+----------------------------------+
```

Unbound floating IPs can be identified by looking at the columns "Fixe IP Address" or "Port". Those fields will be blank if the IP is not bound to a port.

If you don't have any floating IPs available you can just allocate a new one by issuing:

```
~ openstack floating ip create public-network
+---------------------+--------------------------------------+
| Field               | Value                                |
+---------------------+--------------------------------------+
| created_at          | 2022-12-20T09:34:15Z                 |
| description         |                                      |
| dns_domain          | None                                 |
| dns_name            | None                                 |
| fixed_ip_address    | None                                 |
| floating_ip_address | 91.198.2.117                         |
| floating_network_id | 210e26d7-942f-4e17-89b2-571eee87d7e4 |
| id                  | ba086040-8c9e-4590-a3ca-1bfac9102aac |
| name                | 91.198.2.117                         |
| port_details        | None                                 |
| port_id             | None                                 |
| project_id          | f8683d921d4b499eade654f5546c2875     |
| qos_policy_id       | None                                 |
| revision_number     | 0                                    |
| router_id           | None                                 |
| status              | ACTIVE                               |
| subnet_id           | None                                 |
| tags                | []                                   |
| updated_at          | 2022-12-20T09:34:15Z                 |
+---------------------+--------------------------------------+
```

If you want to attach the floating IP to your load balancer then you will first have to obtain the ID of the virtual port of the load balancer.
Issue the following command to do so:

```
~ openstack loadbalancer show my-lb -c vip_port_id
+-------------+--------------------------------------+
| Field       | Value                                |
+-------------+--------------------------------------+
| vip_port_id | 27ebe90b-f0a7-486f-907e-9c579c452ecf |
+-------------+--------------------------------------+
```

Copy the ID from the response and you can now bind the floating ip to your load balancer port with this command:

```
~ openstack floating ip set --port 27ebe90b-f0a7-486f-907e-9c579c452ecf ba086040-8c9e-4590-a3ca-
1bfac9102aac
```

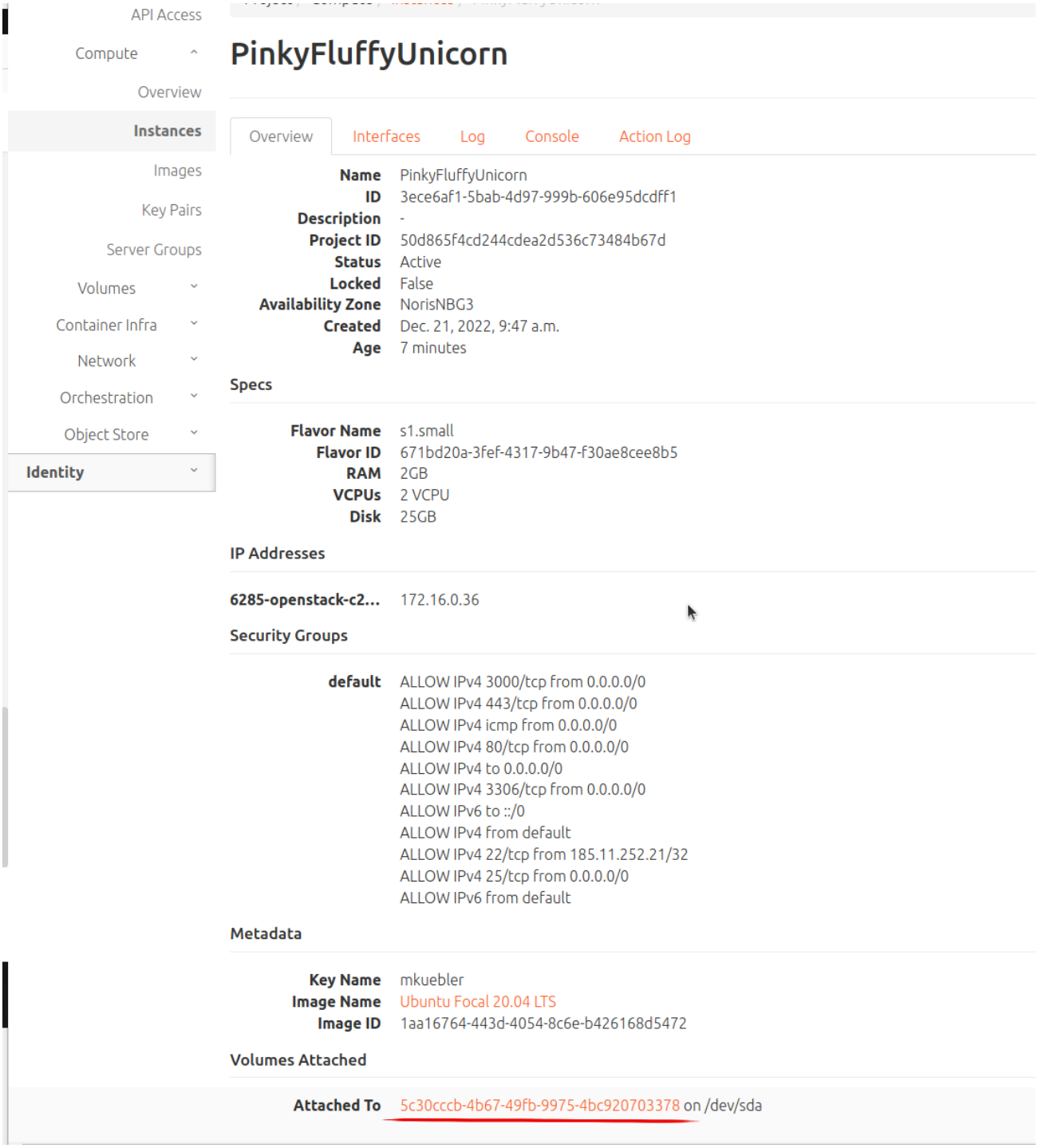
The ID at the "--port" parameter is the one of the vip port. The last ID is the ID of the floating IP you would like to use. Ir this case it was taken from the output of the "openstack floating ip create public-network" command.

This is already everything you need to do for the basic setup of a load balancer.

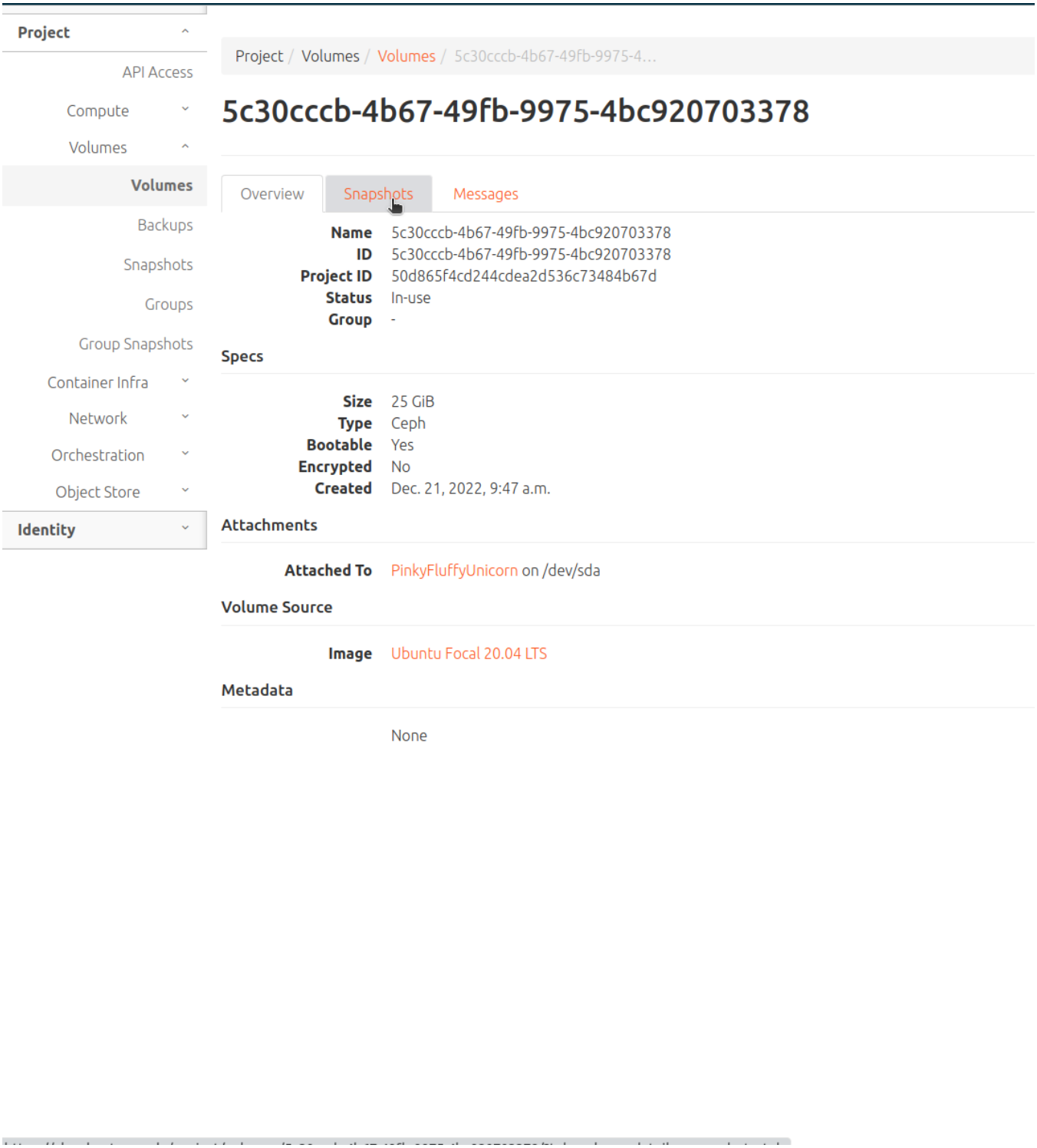
Attaching a snapshot (as an additional volume) to a Ceph-based VM

Locate snapshot

Visit the details page of your VM, click on the VM name, you will see the following screen and the attached disc. Click on the disc to go to the overview page.

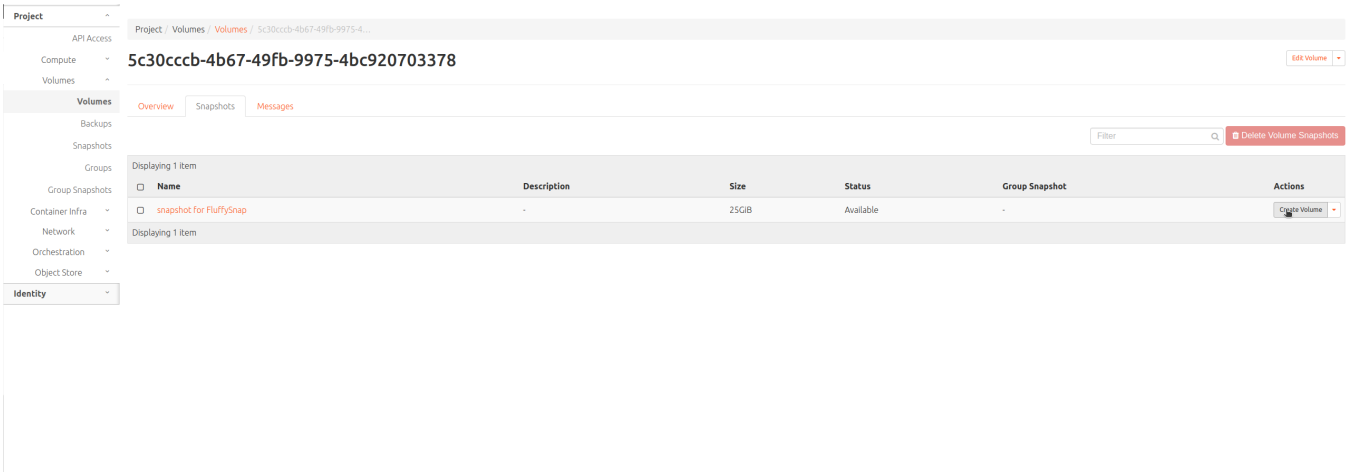


Switch to the snapshots tab to view the snapshots created manually or automatically for this machine.

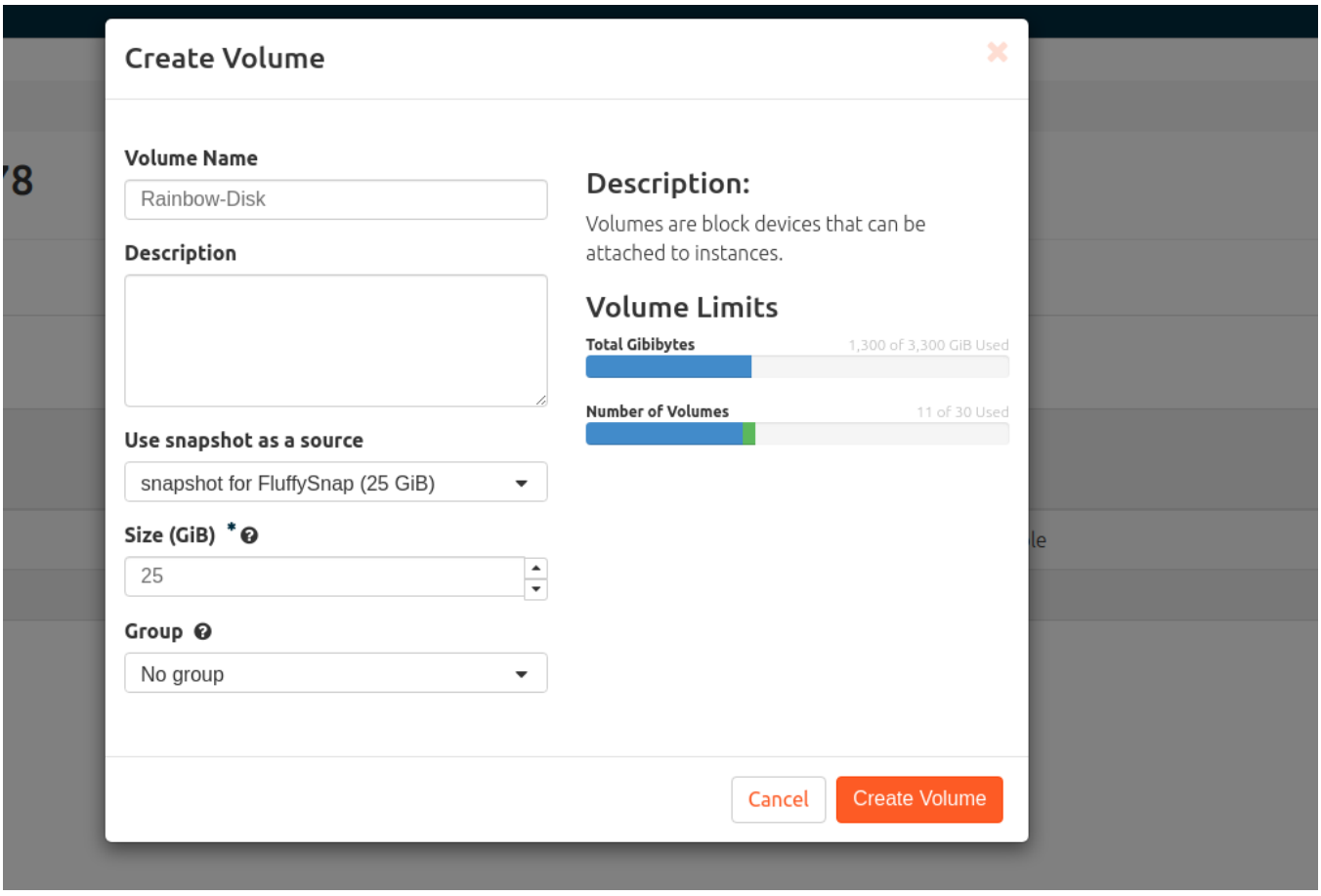


Create volume

On the snapshot overview page, select the desired snapshot and create a volume from it.

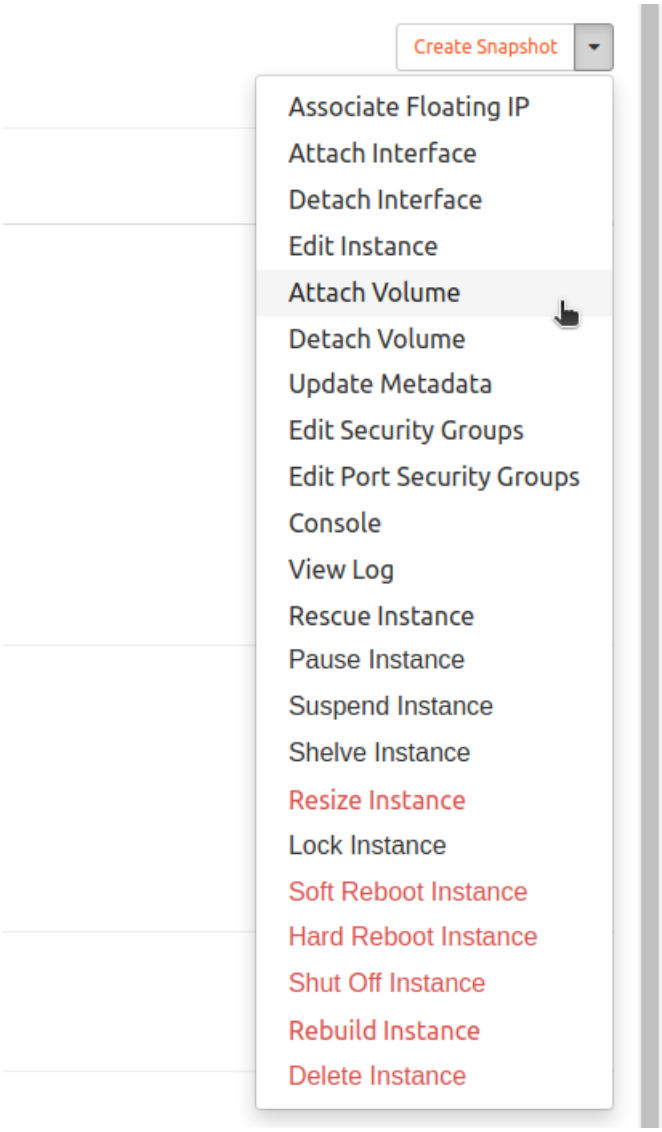


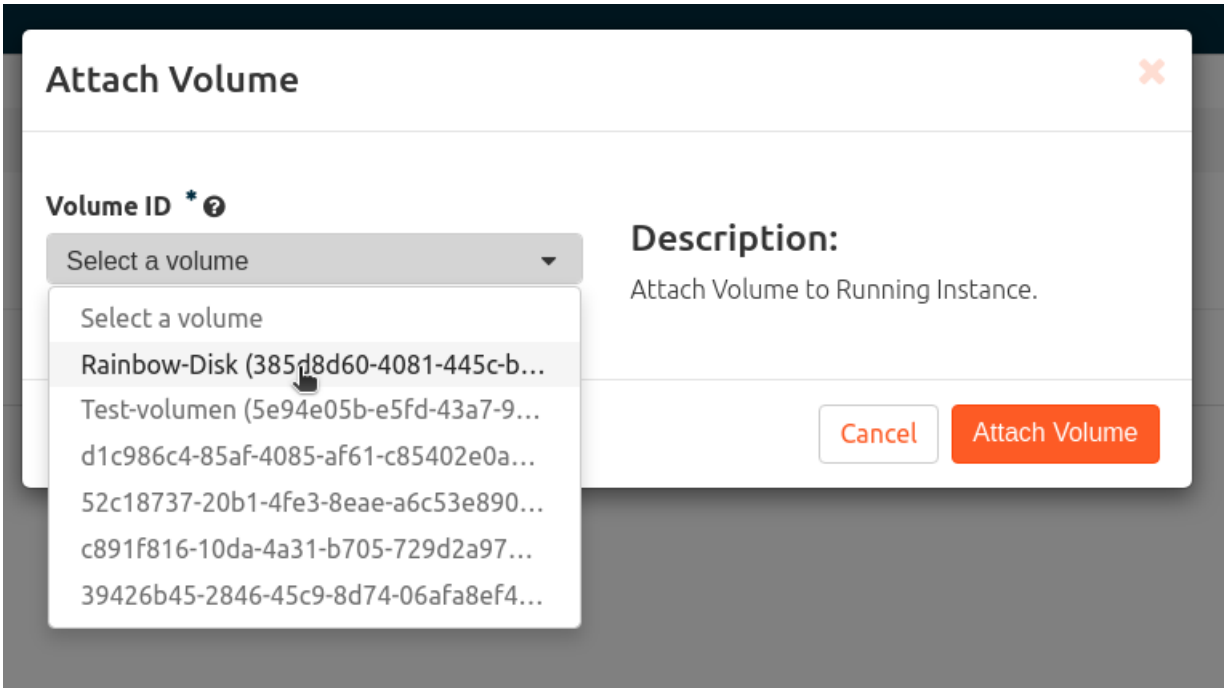
Now assign the desired name for your volume in the next tab.



Attach volume

When the process is complete, you can attach the volume you just created on the overview page of your instances.





Mount volume

Your volume is now ready on the system, now you need to take a few more steps on the system.

Log into the system

1. become root
2. make sure which is the system disc

```
df -h
```

3. Display available system disk

```
fdisk -l
```

4. Disk mounten

```
mount /dev/sdb1 /mnt
```

Disassembly

1. Volume unmounten
   Attention, you must not be in the mount directory

```
umount /mnt
```

2. Detach data carrier at the machine
   Instance overview -> desired instance -> "Detach volume".
   Delete volume and if necessary manually created snapshot
   Volume -> Detach volume -> Delete volume

> If the volume was attached overnight, it may also have been backed up, in which case the snapshot created must be deleted before deleting the volume.

Share networks between multiple OpenStack projects

Please have a look at this Blogpost where we introduced the feature and describe how to use it.

Creating and Using S3 Object Storage

Please have a look at this Blogpost where we introduced the feature and describe how to use it.

Administrative Tasks

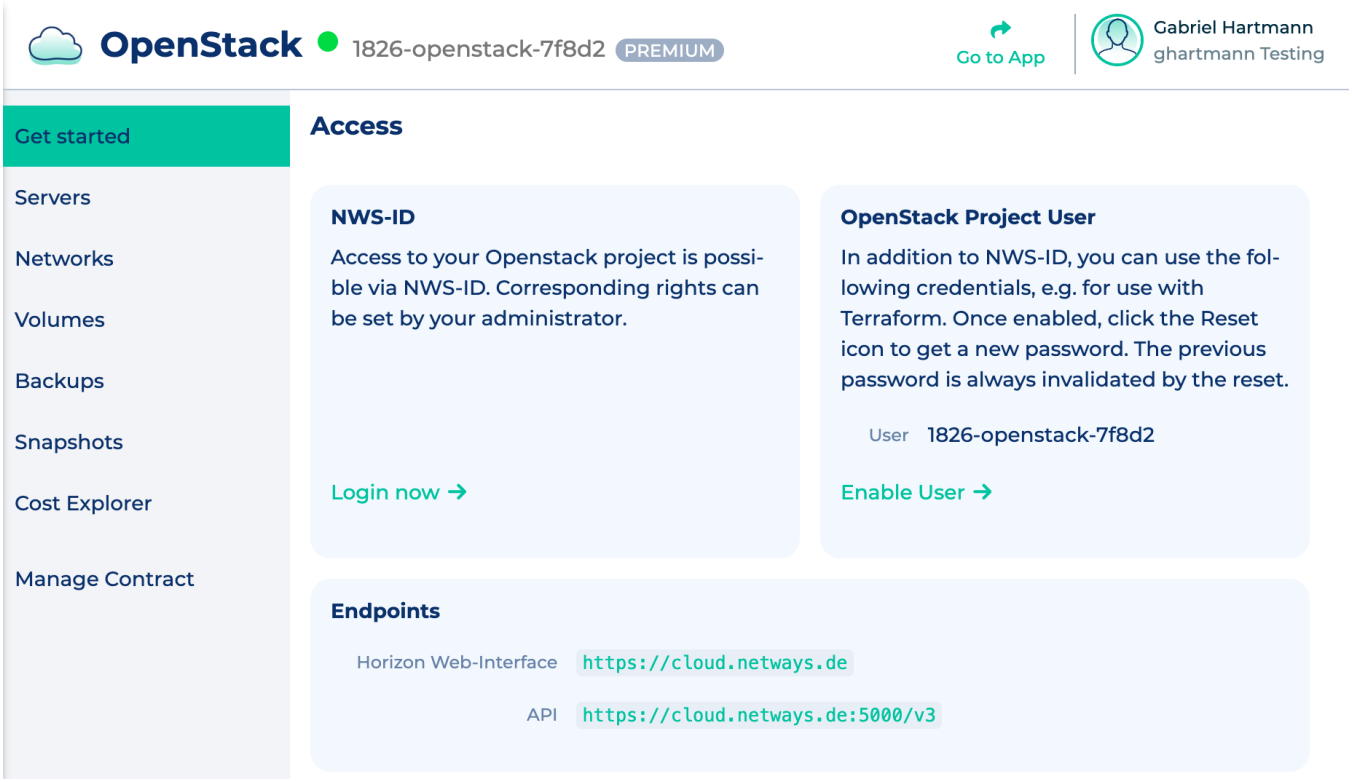
OpenStack Project User

Next to the NWS-ID, the OpenStack Project User provides an additional option to sign in to your OpenStack project. Since NWS-ID comes with some limitations regarding the use of Application Credentials, EC2 Credentials for S3 / Object Storage and Terraform, the credentials of the OpenStack Project User can be used in those scenarios as an alternative.

Enabling the Project User

After creating your Virtual Private Cloud (VPC), the OpenStack Project User will be disabled by default.
So you will have to enable the user first, before you can get the user credentials and make use of the additional user account.

To enable the user, navigate to your VPC. You will find a box with the title "OpenStack Project User", containing an action button "Enable User". Just click on that button to enable the user.



It will take a few seconds to complete the operation. The page will reload automatically once it is finished.

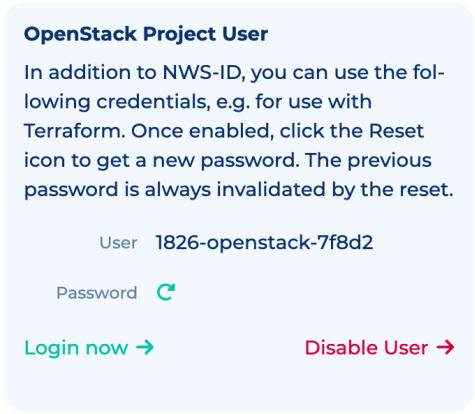
Requesting a new Password

Once the account of the OpenStack Project User has been enabled, you will have the option to request a password.
There are a two **important facts** to know about this action:

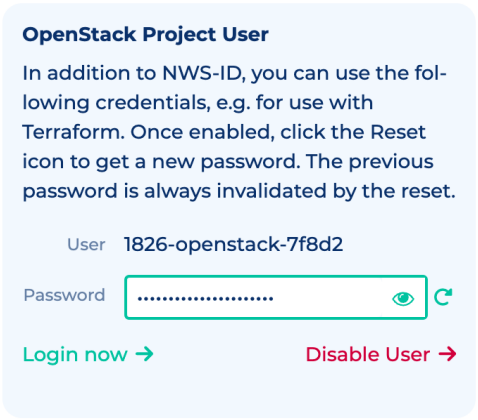
1. **The password will be displayed only once, meaning for as long as you stay on the page**
2. **Requesting a new password will invalidate the previous password**

We therefore advise you to save the password locally in a secure place (like a Password Manager) as soon as you have obtained it.

In order to request a new password, just click on the half-circle-arrow icon next to the "Password" row.

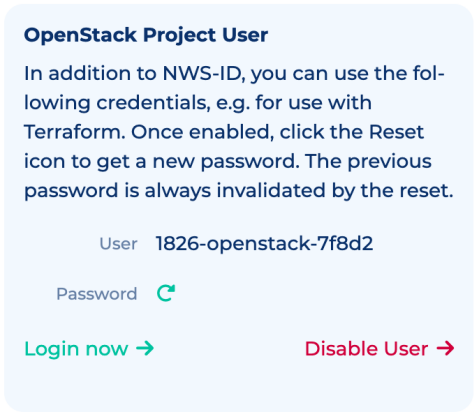


After the action is complete, a password text box appears containing the new password.
By default, the password is hidden. You can view and then copy the password by clicking the eye icon in the password field.



Disabling the Project User

In the OpenStack Project User box, you will find a Disable User button if the user was previously enabled.
After you click the button, the user account is disabled.



Technically, to be precise, the user is not disabled. What happens when you click "Disable User" is that the current password expires with immediate effect and the "Update password" option is locked for the account.

Important note for the use of Application Credentials or EC2 Credentials

Since Application Credentials and EC2 Credentials are always tied to an OpenStack**project and a user**, we recommend using the OpenStack Project User when creating and using these types of credentials.

The problem with using Application Credentials or EC2 Credentials in conjunction with a NWS-ID is that the credentials are only valid as long as the NWS-ID session exists in OpenStack. However, a NWS-ID session in OpenStack expires after one hour. To work around this, one could log back into OpenStack before the session expires. However, this is not a convenient solution and we therefore highly recommend using the OpenStack Project User instead.

Trust for the NWS Customer Interface

The trust allows you to manage your OpenStack resources through the NWS Customer Interface.
You can delete the trust yourself if you do not want our services to interact with your project's resources.
Please note that by deleting the trust, you lose the ability to manage your OpenStack resources through the NWS customer interface. However, this does not affect the Horizon web interface.

Delete a trust

To delete the trust, you must log in to the Horizon Web Interface using the OpenStack project user credentials and then download the OpenStack RC file from the top right menu.
Then, open a terminal and source the OpenStack RC file:

```
source <project-name>-openrc.sh
```

Copy the trust ID from the NWS Customer Interface. Then issue the following command in order to delete the trust:

```
openstack trust delete <trust-id>
```

Create a trust

To create a new trust, log in to the Horizon Web Interface using the OpenStack project user credentials and download the OpenStack RC file from the top right menu.
Open up a terminal and source OpenStack RC file:

```
source <project-name>-openrc.sh
```

Issue the following command to create the trust. The needed IDs can be found in the NWS Customer Interface.

```
openstack trust create --impersonate --role <role-id> --project <project-id> <trustor-id> <trustee-id>
```

Temporary particularities for projects using the new software-defined networking 'OVN'

Our cloud team is conducting a transition to a new software-defined networking (SDN) which is called 'OVN'. Until the transition is complete, our network service is running in parallel mode supporting both SDNs. However, since the previous SDN ('Midonet') is still the default choice for the OpenStack API, there are a few special considerations for projects that are already using OVN.

Please note the following restrictions in OVN projects:

- Floating IPs can take up to 5 minutes to become reachable
- To create networks with terraform Terraform the network_type has to be specified:

```
resource "openstack_networking_network_v2" "net" {
  name = "net"
  segments {
    network_type = "geneve"
  }
}
```

- It is not possible to create Networks via the Horizon UI, instead they should be provisioned via the CLI like so:

```
openstack network create --provider-network-type geneve network1
```

- In an OpenStack project LoadBalancers need to be created with the options "--flavor ovn --availability-zone ovn", Kubernetes clusters will automatically choose the right LoadBalancer flavor and AZ

```
openstack loadbalancer create --vip-subnet-id mysubnet --name lb1 --flavor ovn --availability-zone ovn
```

Creating LUKS Encrypted Volumes

OpenStack allows you to encrypt volumes **at rest** using LUKS. On NWS OpenStack, the **volume type** that provides encryption-at-rest is conveniently called **LUKS**. See below for further information and how to setup encrypted volumes.

How Does OpenStack Encrypt Volumes?

When you create an encrypted volume from the Horizon UI or the OpenStack CLI, OpenStack will generate the necessary **encryption key** for you and store it in Barbican, OpenStack's **key manager**. Alll LUKS encrytion keys on NWS OpenStack use **256bit aes-xts-plain64 encryption.**

> As **Secrets** are billed on a per-secret basis in NWS OpenStack, each encrypted volume will incu**minimal additional cost** (0.40€ p. Secret p. Month, 03/2024)

When **attaching** an encrypted volume to a server, OpenStack will decrypt it in the background using the encryption key stored in Barbican. The attached volume can be handled and consumed on the server like any other (unencrypted) volume.

Creating Encrypted Volumes in OpenStack

Encrypted volumes can be created either from the Horizon UI or from the terminal using the OpenStack CLI.

Using Horizon UI

In your OpenStack Horizon UI (at https://cloud.netways.de) make sure to pick the correct **project,** then follow the steps below:

1. Navigate to **Volumes > Volumes**
2. Click on **Create Volume**
3. **Configure** the volume
   1. [**Required**] Set a **name**
   2. [**Optional**] Set a **description**
   3. [**Required**] Pick a **volume source**
   4. [**Required**] Set **LUKS** as **Volume Type**
   5. [**Optional**] Adjust **volume size**
   6. [**Optional**] Pick a **volume group**
4. Confirm settings and create volume by clicking **Create Volume**

OpenStack will start creating the volume, which will appear listed under **Volumes > Volumes** upon reload.

Using OpenStack CLI

Source your `OpenStackRC.sh` file and pick the correct **project**:

```
source nws-id-openstack-rc.sh

Testing authentication and fetching project list ...

Please select one of your OpenStack projects.

1) 20631-openstack-04223  3) 5475-openstack-41b6b   5) 5475-openstack-8bdaf   7) 5475-openstack-a169e
2) 5475-openstack-1ccca   4) 5475-openstack-4745f   6) 5475-openstack-9d52e   8) 5475-openstack-c716d
Enter a number: 1
Selected project: 20631-openstack-04223
```

Then, create a new, encrypted volume like this:

```
openstack volume create \
  --type LUKS \
  --size <size> \
  --description <description> \
  --image <image> \
  <name>
```

The following options can be passed as parameters:

- `<size>`: **size** of the volume in GB, **required** if `<image>` is **not** passed
- `<description>`: **description** of the volume, **optional**
- `<image>`: **image reference** if the volume should be based on an image,**optional**
- `<name>`: **name** of the volume, **required**

Attaching Encrypted Volumes to Servers

Encrypted volumes can be attached to servers either from the Horizon UI or from the terminal using the OpenStack CLI, just like any other volume.

Using Horizon UI

In your OpenStack Horizon UI (at https://cloud.netways.de) make sure to pick the correct **project,** then follow the steps below:

1. Navigate to **Compute > Instances**
2. **Identify** the server you want to attach the volume to in the list
3. **Expand** the dropdown menu in the server's **Action** column
4. Click **Attach Volume** and select the **desired volume**
5. Confirm by clicking **Attach Volume**.

After a few seconds, the volume should be visible on the server at `/dev/sdX`

Using OpenStack CLI

Source your `OpenStackRC.sh` file and pick the correct **project**:

```
source nws-id-openstack-rc.sh

Testing authentication and fetching project list ...

Please select one of your OpenStack projects.

1) 20631-openstack-04223  3) 5475-openstack-41b6b   5) 5475-openstack-8bdaf   7) 5475-openstack-a169e
2) 5475-openstack-1ccca   4) 5475-openstack-4745f   6) 5475-openstack-9d52e   8) 5475-openstack-c716d
Enter a number: 1
Selected project: 20631-openstack-04223
```

Then, attach the volume to a server like this:

```
openstack server add volume --device <device> <server> <volume>
```

The following options need to be passed as parameters:

- `<device>`: server internal **device name**, e.g. `/dev/sdb`, **required**
- `<server>`: **name** or **UUID** of the server to attach the volume to, **required**
- `<volume>`: **name** or **UUID** of the volume to attach to the server, **required**

After a few seconds, the volume should be visible on the server at `/dev/sdX`

Further Information

If you are looking for more detailed instructions on creating and mounting encrypted volumes or want to learn more about encrypted volumes on OpenStack in general, see the links below:

- **NWS-Blogpost**: LUKS Encrypted Storage on OpenStack
- **OpenStack Documentation:** Volume encryption supported by the key manager

Related Links

On our tutorials page you will find numerous articles about handling our OpenStack service