

---

# **stfc-cloud-docs Documentation**

***Release 1.0***

**CloudTeam**

**Oct 11, 2023**



**GETTING STARTED:**

<b>1</b>	<b>Common Commands</b>	<b>3</b>
<b>2</b>	<b>Authentication</b>	<b>5</b>
<b>3</b>	<b>How-To Articles</b>	<b>9</b>
<b>4</b>	<b>Policies</b>	<b>109</b>
<b>5</b>	<b>Contact Details</b>	<b>113</b>
<b>6</b>	<b>FAQS</b>	<b>115</b>
<b>7</b>	<b>Fault Fixes</b>	<b>123</b>
<b>8</b>	<b>Flavors</b>	<b>125</b>
<b>9</b>	<b>Image Types</b>	<b>127</b>
<b>10</b>	<b>Reference Documents</b>	<b>129</b>
<b>11</b>	<b>Aodh and Gnocchi</b>	<b>147</b>
<b>12</b>	<b>Kubernetes</b>	<b>167</b>
<b>13</b>	<b>Improving this Documentation</b>	<b>175</b>
<b>14</b>	<b>Heat</b>	<b>177</b>
<b>15</b>	<b>Manila</b>	<b>201</b>
<b>16</b>	<b>Octavia</b>	<b>221</b>
<b>17</b>	<b>Swift</b>	<b>247</b>
<b>18</b>	<b>Magnum</b>	<b>281</b>
<b>19</b>	<b>Legacy Documentation</b>	<b>317</b>
<b>20</b>	<b>Indices and tables</b>	<b>323</b>



**Warning:** New Confluence

We are in the process of migrating this document repository to our new confluence: <https://stfc.atlassian.net/wiki/spaces/CLOUDKB/overview> so pages here will be updated with a link to the equivalent document in the new space.



## COMMON COMMANDS

On Train, Hypervisors often disable themselves and remove their services from the pool of resources if there are 10 build failures in a row on a hypervisor.

### 1.1 List available images

To list available images do:

```
openstack image list
```

### 1.2 List available flavors

To list available flavors run:

```
openstack flavor list
```

### 1.3 List available networks

To list available networks run:

```
openstack network list
```

### 1.4 List running servers

Showing all hosts in your current projects:

```
openstack server list
```

## 1.5 Show a specific server

Looking at a specific server:-

```
openstack server show <server ID>
```

## AUTHENTICATION

### 2.1 Application Credentials

Application credentials can be used to allow applications or other users certain access to specific projects without embedding the main user's password in the application configuration.

Application credentials can be created either through the web interface or command line

---

**Note:** Here we assume that users are familiar with using `clouds.yaml`. See [Setting Up Clouds.yaml](#) for more information.

---

**Warning:** We **strongly** recommend that any application credentials created have a lifetime greater than one hour.

#### 2.1.1 Create Credential from the Web interface

The option for generating application credentials can be found under *Identity -> Application Credentials*. Select *Create Application Credential*.

The currently selected project will be used to generate application credentials for the user and project pair. If you have access to multiple projects, the project these credentials are valid for can be changed by selecting the project from the drop down menu at the top of the page.

The form has 7 fields:

- **Name:** The name that is given to the application credential.
- **Description:** Description of the application credential. This could be the description of which application will use this credential or the purpose of the credential.
- **Secret:** A password for the credential which will be used in order to authenticate access through keystone. We recommend you leave this blank or use a tool like *pwgen* to generate this automatically
- **Expiration Date:** Date that the application credential expires and will be deleted. If left blank, the application credential will not expire.
- **Expiration Time:** The time the credential expires, if a date is given by default the time is 00:00:00.
- **Roles:** The role that the application credential is authorized to have.
- **Unrestricted Option:** By default this is left unticked.

**Warning:** If ticked, the application credential will be able to create additional application credentials.

**For security, always have the application credential restricted.** Restrictions on these operations are deliberately imposed as a safeguard to prevent a compromised application credential from regenerating itself.

It is important to make a note of the secret which has been chosen for the application credential as it is only revealed once after the application credential has been generated. From the web interface, the RC file and the `cloud.yaml` file is available once and it is recommended to download these files. If the user has forgotten the secret, a new application credential will have to be created.

## 2.1.2 Create Credential from Command Line

To view the list of application credentials, we can use the command:

```
openstack application credential list
```

This will return an empty line if no application credentials have been created yet, or a table similar to the one below:

```

+-----+-----+-----+-----+
↪-----+
↪-----+
| ID              | Name              | Project ID        | Expires At
↪Description      |
↪
+-----+-----+-----+-----+
↪-----+
↪-----+
| APPLICATION_CREDENTIAL_ID_1 | Test-App-Credential-1 | PROJECT_ID        | This is
↪a test application credential generated using the web interface. | 2020-07-01T00:00:00.
↪000000 |
| APPLICATION_CREDENTIAL_ID_2 | Test-app-credential-2 | PROJECT_ID        | Test
↪Application Credential from command line. | None
↪
+-----+-----+-----+-----+
↪-----+
↪-----+

```

To create a new application credential, you can use the command:

```
openstack application credential create [-h]
                                         [-f {json,shell,table,value,yaml}]
                                         [-c COLUMN] [--noindent]
                                         [--prefix PREFIX]
                                         [--max-width <integer>]
                                         [--fit-width] [--print-empty]
                                         [--secret <secret>]
                                         [--role <role>]
                                         [--expiration <expiration>]
                                         [--description <description>]
                                         [--unrestricted] [--restricted]
                                         <name>
```

Below is an example of creating an application credential which expires on 06/07/2020 at 00:00:00 and the secret has been generated automatically by OpenStack.

```
openstack application credential create --expiration 2020-07-08T00:00:00 --description
↳ "Example Application Credential" Example-Credential
```

+-----+		
↳ +-----+		
Field	Value	
↳ +-----+		
description	Example Application Credential	↳
expires_at	2020-07-08T00:00:00.000000	↳
id	APPLICATION_CREDENTIAL_ID	↳
name	Example-Credential	↳
project_id	PROJECT_ID	↳
roles	user	↳
secret	SECRET	↳
system	None	↳
unrestricted	False	↳
user_id	USER_ID	↳
+-----+		
↳ +-----+		

**Note:** The secret is only revealed **once**. If a user has forgotten the secret, a new application credential as to be generated.

After an application credential has expired, it is still visible in the application credential list. If the application credential is used after it has expired, nothing will happen and no one can get access to the project via the expired credential.

### 2.1.3 RC source and clouds.yaml file

Unlike in the web interface, the RC file and the clouds.yaml file are not automatically generated. They need to be created separately by the user. The following are examples of a clouds.yaml file and RC file for an application credential.

#### clouds.yaml

```
# This is a clouds.yaml file, which can be used by OpenStack tools as a source
# of configuration on how to connect to a cloud. If this is your only cloud,
# just put this file in ~/.config/openstack/clouds.yaml and tools like
# python-openstackclient will just work with no further config. (You will need
# to add your password to the auth section)
# If you have more than one cloud account, add the cloud entry to the clouds
# section of your existing file and you can refer to them by name with
# OS_CLOUD=openstack or --os-cloud=openstack
clouds:
  openstack:
    auth:
      auth_url: AUTH_URL
      application_credential_id: "APP_CREDENTIAL_ID"
      application_credential_secret: "APP_CREDENTIAL_SECRET"
    region_name: "RegionOne"
    interface: "public"
    identity_api_version: 3
    auth_type: "v3applicationcredential"
```

#### RC File

```
#!/usr/bin/env bash
export OS_AUTH_TYPE=v3applicationcredential
export OS_AUTH_URL=https://AUTH-URL #this will be the Identity service endpoint URL
↪under API Access
export OS_IDENTITY_API_VERSION=3
export OS_REGION_NAME="RegionOne"
export OS_INTERFACE=public
export OS_APPLICATION_CREDENTIAL_ID=APP_CREDENTIAL_ID
export OS_APPLICATION_CREDENTIAL_SECRET=APP_CREDENTIAL_SECRET
```

### 2.1.4 References

[https://docs.openstack.org/keystone/latest/user/application\\_credentials.html](https://docs.openstack.org/keystone/latest/user/application_credentials.html)      <https://docs.openstack.org/api-ref/identity/v3/index.html?expanded=authenticating-with-an-application-credential-detail#application-credentials>  
<https://cloud.garr.it/compute/app-credential/>      [https://access.redhat.com/documentation/en-us/red\\_hat\\_openstack\\_platform/14/html/users\\_and\\_identity\\_management\\_guide/application\\_credentials](https://access.redhat.com/documentation/en-us/red_hat_openstack_platform/14/html/users_and_identity_management_guide/application_credentials)

## HOW-TO ARTICLES

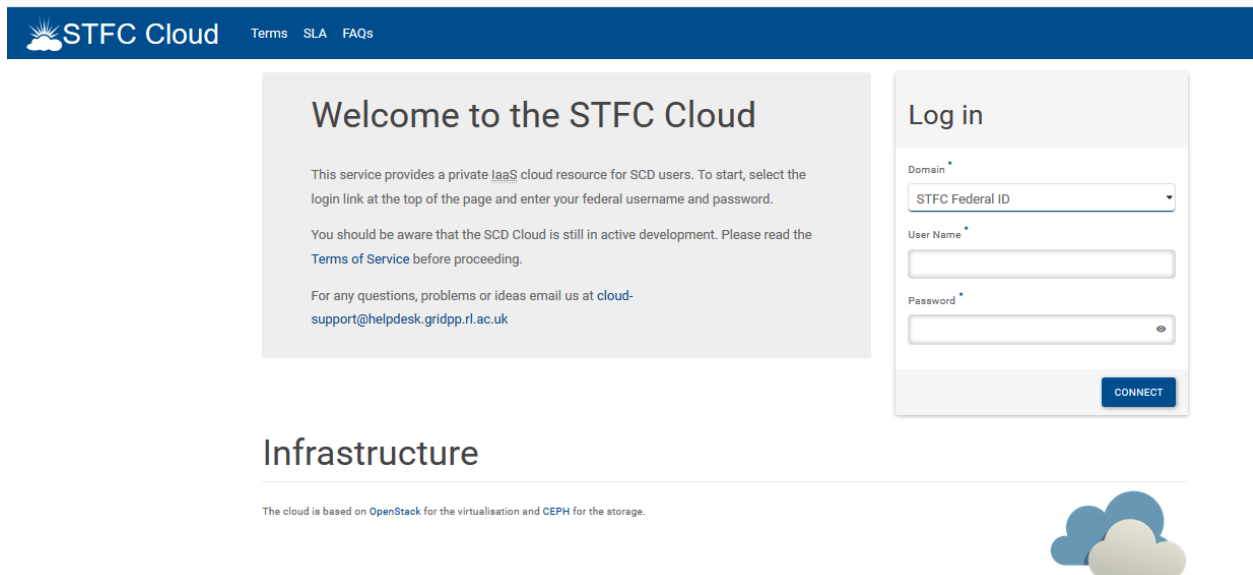
### 3.1 Adding Your SSH Key into the OpenStack Web Interface

#### 3.1.1 Overview

This document describes how to add your public ssh key into the openstack ssh web user interface (<https://openstack.stfc.ac.uk>). It assumes you have already contacted [cloud-support@stfc.ac.uk](mailto:cloud-support@stfc.ac.uk) to either initiate a project, or to add user accounts to an existing project. It also assumes you already have an SSH key pair generated, and you just wish to add your existing public ssh key. You need to import your ssh public key into the Openstack web interface as then the system will inject your ssh public key (and username) into any Virtual machine that you create.

#### 3.1.2 Procedure

Login to the web site: <https://openstack.stfc.ac.uk> using the credentials you have been given. If you have a STFC FEDID account, make sure that you set the “Domain” to be “STFC Federal ID”:-



The screenshot shows the STFC Cloud web interface. At the top is a dark blue header with the STFC Cloud logo and links for Terms, SLA, and FAQs. The main content area is divided into two columns. The left column has a light gray background and contains a 'Welcome to the STFC Cloud' message, a paragraph about the service, a note about active development, and contact information. The right column has a white background and contains a 'Log in' form with fields for Domain (a dropdown menu showing 'STFC Federal ID'), User Name, and Password, followed by a 'CONNECT' button. Below the login form is a section titled 'Infrastructure' with a small text line and a cloud icon.

#### AddingSSHKeysInWebInterface

Once logged in, click on the Left Hand Side “Compute” option, and then “Key Pairs”. You should see a screen similar to below, with the exception that if this is your first time connecting in, no key pairs will be listed:-

If you do not already have an SSH key pair, you can use the “Create key pair” button. If you have an existing ssh key pair, you should click on the “Import Public Key” button. On doing so, a pop-up window will appear:-

From here, you can cut and paste your “id\_rsa.pub” key (often found on a linux host you already have ssh access to, in the ~/.ssh directory – you can run “cat authorized\_keys” to display the key and then cut and paste it into the web-browser window. Give the key pair a name as well. NOTE: Be *very* careful with pasting the key, or importing it via the file “Browse” button: SSH keys are very sensitive to any of the characters being incorrect, including line feed characters – the key should be one continuous line, without spaces, Carriage Returns or Line Feed characters

## 3.2 Adding multiple user accounts to a VM

### 3.2.1 Outline

This script will add a list of fed IDs or openstack account names to the VM it is run on. Make sure that all the accounts you are attempting to add exist within the Openstack project the VM is in.

### 3.2.2 Method

Here is the script. Save this as *addusers-ubuntu.sh* to the VM.

```
#!/bin/bash

FEDIDS=$(cat $1)
for fedID in $FEDIDS
do
    $(echo -e "\n\n" | adduser $fedID -q --gecos ",,," ) &>/dev/null
done
```

To run this script, first create a file called *fedIDs.txt* containing the fed IDs you want to add as user accounts. Separate each ID with a new line. Then, run the script with `sudo` (you will need `sudo` privileges to add users) as follows, passing the file containing the fedIDs as an argument:

```
sudo bash addusers-ubuntu.sh fedIDs.txt
```

To check that the users have been added correctly, you can check the contents of the */etc/passwd* file. You should see the new accounts at the bottom of this file, e.g.

```
$ tail -n 1 /etc/passwd
<fedID>:x:1008:1006:,,,:/home/<fedID>:/bin/bash
```

## 3.3 Adding a user account to multiple cloud hosts within a project

### 3.3.1 Outline

For some types of Openstack projects, it's common to have a list of user accounts, that may or may not be external users, that need ssh access to a (large) number of hosts within that project. This document provides an outline approach as to how to approach this task.

### 3.3.2 Method

Make sure you have root access to the systems, or at the very least, have a login to each system with `sudo` access. If you are using a ssh key pair to login (the preferred method within STFC), make sure you have your private key loaded on a ssh-agent and ensure that "Agent forwarding" feature is enabled. If this is a common task that you have to do, it is worth using the same "command and control" hosts in all of your operations for running scripts on the hosts within projects. Step 1 – Find the list of hosts within a project Find the project from: `Openstack project list | grep -v 844e | grep -v rally ...` from there, you can list all of the virtual machines that are in a particular project:- `Openstack server list --project "Rucio at RAL"` This will return a list of hostnames and their IP addresses (and float IP addresses). You can make this just a list of IP addresses using something like:- `openstack server list --project "Rucio at RAL" | awk -F|`

`{ print $5 } | grep 'Internal' | sed 's/^ Internal=/' ...` which just returns a list of the 172.16.x.y network IP addresses – there is no point in returning the 192.168.x.y IP addresses as you cannot directly ssh to these anyway, but any float IP addresses, you could append as needed. Output this to a file “all\_IP\_addresses.text”. Check login to each host, identify host and check which hosts have users enabled on them

## 3.4 Associate a floating IP to a VM

Assumes that you have float-IPs associated with your project. If you do not - contact Cloud Support.

In the Horizon Web user interface (<https://Openstack.stfc.ac.uk>) in the Left Hand Side Menu, select:-

Compute->Instances

Find the VM (that has a Private 192.168.x.y IP address) that you wish to associate a Floating IP address with. On the Right Hand side, click on the down arrow, and select “ASSOCIATE FLOATING IP” - a pop-up window will appear. Click on “Select an IP address” - a drop-down list of available floating IP addresses will be listed. Click on one. Click on the “ASSOCIATE” button. You should be returned to the “Instances” list, and you should now see your float-IP associated with your VM, as well as its original 192.168.x.y address.

For the equivalent using the openstack API command line:-

To find float-IP addresses associated with your project, use the command:-

```
openstack floating ip list --project "project name"
```

... where project name is your project name in quotations. This will return a list of float IP addresses, and will show which ones are in use, and which ones are not. Take note of one of the float IP addresses that is free. Now list the VM instances in your project, use the command:-

```
openstack server list --project "project name"
```

... will list all of your VM instances. Take note of the first column, which is the instance ID of your chosen VM. To associate a floating IP selected earlier, use the command:-

```
openstack server add floating ip <VM instance ID> <Floating IP>
```

This should then result with the floating IP address being associated with the floating IP names. You should now be able to ssh directly into the VM, using the floating IP address as a target, assuming that the security group allows ssh inbound, and any additional firewalls in the way, also have ssh allowed inbound.

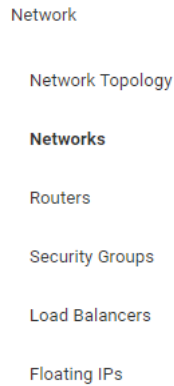
## 3.5 Changing the DNS server for a Magnum Cluster

- *Via the Web Interface*
- *Via the Command Line*

The default DNS server for a cluster generated by Magnum is 8.8.8.8, which is the Google Public DNS. You may wish to use your own DNS server, follow the steps below to update the DNS on the cluster subnet.

### 3.5.1 Via the Web Interface

- Access the networks tab on the web interface.



- Select the network for your cluster. It should be named the same as your cluster
- Go to the subnets tab
- Select edit subnet

Network Address	IP Version	Gateway IP	Actions
10.0.0.0/24	IPv4	10.0.0.1	EDIT SUBNET ▾

- Select “Subnet Details”

## Edit Subnet

×

Subnet \*

Subnet Details

Subnet Name

Update a subnet associated with the network. Advanced configuration are available at "Subnet Details" tab.

Network Address ⓘ

Gateway IP \* ⓘ

☐ Disable Gateway

CANCEL

« BACK

NEXT »

- Update the IP in DNS Name Servers to the IP of the DNS you wish to use and click save

# Edit Subnet



Subnet

Subnet Details

☒ Enable DHCP

Specify additional attributes for the subnet.

Allocation Pools

10.0.0.2,10.0.0.254

DNS Name Servers

8.8.8.8

Host Routes

CANCEL

« BACK

SAVE

## 3.5.2 Via the Command Line

- Get a list of your networks with

```
openstack network list
```

- Get the subnets in the network with the same name as your cluster

```
openstack subnet list --network <Name or ID of network>
```

- If you want to add a new DNS to the subnet, and have multiple possible DNS servers run the following command

```
openstack subnet set --dns-nameserver <Additional DNS IP here> <Subnet name or ID>
```

- If you want to replace the existing DNS with your own run the following command

```
openstack subnet set --no-dns-nameserver --dns-nameserver <DNS IP here> <Subnet name or ID>
```

The `--no-dns-nameserver` flag will remove the existing DNS IPs from the subnet before you add the new one. This ensures that there are not multiple DNS servers on your subnet.

- You can add multiple DNS servers at once by repeating the flag. The following command will erase the existing DNS server and add two new ones.

```
openstack subnet set --no-dns-nameserver --dns-nameserver <DNS IP here> --dns-nameserver
↔<Additional DNS IP here> <Subnet name or ID>
```

## 3.6 Configuring SSH Agent Forwarding in Windows Subsystem for Linux

Configuring WSL is pretty easy, this link offers a good tutorial.

<https://www.windowcentral.com/install-windows-subsystem-linux-windows-10>

Ensure your distribution of Linux in WSL is fully up to date (some versions have known issues around agent forwarding).

I used Ubuntu 18.04 LTS so I ran:

```
sudo apt-get update
sudo apt-get upgrade
```

This left me with a nice up to date install. You can then run the following commands to start and add your key to the ssh agent:

```
eval $(ssh-agent -s)
ssh-add <path-to-your-ssh-private-key>
```

Then you can ssh to your host with agent forwarding using

```
ssh -A <your-remote-host>
```

Then you can SSH on from there.

## 3.7 Configuring an X Server in Windows Subsystem for Linux

First select a Windows based X server such as Xming or VcXsrv

I chose to use VcXsrv, download and install this. At time of writing the default installation options work fine. Launch VcXsrv by running XLaunch

Open your WSL window and run the following command to set your default display:

```
export DISPLAY=localhost:0.0
```

You should then be able run X applications

You can test this by running

```
sudo apt install x11-apps
```

then

```
xclock
```

You should then see a little graphical clock appear somewhere on your screen.

To be able to use this remotely you will need to generate an XAuthority file. This can be done with the following:

```
xauth generate :0 . trusted
```

You should then be able to ssh into a remote host with the following command provided that the remote host is appropriately configured :

```
ssh -X <user>@<your.remote.host>
```

## 3.8 Create a Heat stack

Heat is an orchestration component of OpenStack which allows templates to be used to define resources and to specify the relationships between resources in a stack. This page explains the basics of how to create a heat stack. For more detailed docs and examples, see the **Heat Advanced Material** pages at <https://stfc-cloud-docs.readthedocs.io/en/latest/Heat/index.html>.

To check Heat is available we can run the command:

```
openstack stack list
```

This command returns the list of stacks in the user's project. If no stacks have been created, this command returns an empty line. If Heat is not installed, see the more detailed pages on Heat: <https://stfc-cloud-docs.readthedocs.io/en/latest/Heat/IntroducingOpenStackHeat.html#heat-commands>

To create a stack, we first need to write a template which will be used to launch the stack.

### 3.8.1 Templates

Templates are used to define the resources which should be created as part of a stack and the relationships between resources. With the integration of other OpenStack components, Heat Templates allow the create of most OpenStack resource types. This includes infrastructure resources such as instances, databases, security groups, users, etc. The Heat Orchestration Template (HOT) is native to OpenStack for creating stacks. These templates are written in YAML.

#### Heat Orchestration Templates

The structure of a Heat Orchestration Template is given as follows:

```
heat_template_version: 2018-08-31 #OpenStack Version we want to use.
                                #Here, we want to use template for the Rocky release.
↪ onwards

description: #description of the template

parameter_groups: #declares the parameter group and order.
                  #This is not a compulsory section, however it is useful for grouping
                  #parameters together when building more complex templates.
```

(continues on next page)

(continued from previous page)

```

parameters: #declares the parameters for resources

resources: #declares the template resources
# e.g. alarms, floating IPs, instances etc.

outputs: #declares the output of the stack

conditions: #declares any conditions on the stack

```

### heat\_template\_version

This tells heat which format and features will be supported when creating the stack.

For example:

```
heat_template_version: 2018-03-02
```

Indicates that the Heat Template contains features which have been added and/or removed up to the Queens release of OpenStack.

The template version:

```
heat_template_version: 2018-08-31
```

Indicates that the Heat Template contains features which have been added and/or removed up to the Rocky release of OpenStack. For templates being used in the current release of OpenStack, heat uses Queen's templates is used.

A list of template versions can be found here: [https://docs.openstack.org/heat/train/template\\_guide/hot\\_spec.html#heat-template-version](https://docs.openstack.org/heat/train/template_guide/hot_spec.html#heat-template-version)

### parameter\_groups

Parameter groups indicate how the input parameters are grouped. The order of the parameters are given in lists. This section is not strictly compulsory for launching stacks, however it is useful for more complex templates where there are several input parameters to be used.

The syntax for parameter\_groups is:

```

parameter_groups:
- label: # label defining the group of parameters
  description: #description of parameter group
  parameters:
    - <parameter-1> #name of first parameter
    - <parameter-2> #name of second parameter
    - <parameter-3> #name of third parameter

```

## Parameters

The parameters section specifies the input parameters that have to be provided when the template is initialised. The syntax for parameters is of the form:

```
parameters:
  <param name>:
    type: <string | number | json | comma_delimited_list | boolean>
    label: <human-readable name of the parameter> #optional
    description: <description of the parameter> #optional
    default: <default value for parameter> #optional - this is used if the user does not_
    ↪ specify a value.
    hidden: <true | false> #default option is false - this determines whether the_
    ↪ parameter is hidden from the user if the user requests information about the stack.
    constraints: <parameter constraints> #optional - list of constraints to apply to the_
    ↪ parameter. The stack will fail if the parameter values do not comply to the_
    ↪ constraints.
    immutable: <true | false> #default is false - this determines whether a parameter is_
    ↪ updateable after the stack is running.
    tags: <list of parameter categories> #optional input - list of strings to specify_
    ↪ the category of the parameter.
```

## Resources

This is a compulsory section and must contain at least one resource. This could be an instance, floating IP, Network, key pair, etc.

A list of the different OpenStack resources which can be used in a Heat template can be found here: [https://docs.openstack.org/heat/latest/template\\_guide/openstack.html](https://docs.openstack.org/heat/latest/template_guide/openstack.html)

```
resources:
  <resource ID>: #must be unique within the resources section of the template.
    type: <resource type> #e.g OS::Nova::Server, OS::Nova::Port,_
    ↪ OS::Neutron::FloatingIPAssociation, etc.
    properties: #list of resource-specific properties that can be provided in place or_
    ↪ via a function.
      <property name>: <property value>
      metadata: #optional
      <resource specific metadata>
      depends_on: <resource ID or list of ID> #optional
      update_policy: <update policy> #optional - this is given in the form of a nested_
      ↪ dictionary.
      deletion_policy: <deletion policy> #optional - allowed deletion policies are_
      ↪ Delete, Retain, and Snapshot
      external_id: <external resource ID> #optional - can define a resource which is_
      ↪ external to the stack
      condition: <condition name or expression or boolean> #optional input - decides_
      ↪ whether the resource should be created based on a given condition
```

Below is an example of the resource section for an instance.

```
my_instance: #name of the instance
  type: OS::Nova::Server
```

(continues on next page)

(continued from previous page)

```

properties:
  image: image_id #retrieves the image ID from image_id parameter
  flavor: flavor_id #retrieves the flavor ID from flavor_id parameter
  key_name: key_name #retrieves the key pair from key_name parameter
  networks:
    - network: network_name #define the internal network as Internal
  security_groups:
    - security_group_id

```

## Outputs

Outputs define the parameters that should be available to the user after a stack has been created. This would be, for example, parameters such as the IP addresses of deployed instances, or the URL of web applications deployed as a stack. Each output is defined as a separate block within the outputs section:

```

outputs:
  <parameter name>:
    description: <description>
    value: <parameter value>
    condition: <condition name or expression or boolean>

```

## Conditions

The conditions section in the heat template defines at least one condition that is evaluated based on the input parameter values when a user creates or updates a stack. The conditions can be associated with resources, the properties of the resources and the output.

The syntax for conditions in the heat template is given by:

```

conditions:
  <condition_name_1>: {expression_1}
  <condition_name_2>: {expression_2}

```

### 3.8.2 Example Template

The following template (example-template.yaml) is for a stack containing a single instance.

```

heat_template_version: 2018-08-31 #OpenStack Rocky Version

description: An example template which launches instances.

parameter_groups: # Optional - helps to group parameters together
  - label: Instance parameters #human-readable label defining the associated group of
    ↪ parameters
    description: The parameters which are required to launch an instance. #description
    ↪ of parameter group
    parameters: #Parameters are given same order as launching an instance using
    ↪ openstack server create command
    - key_name #name of keypair to SSH into instance

```

(continues on next page)

(continued from previous page)

```

- image_id #name can be used as well, but it's better practice to use ID
- flavor_id #name or ID, though it is better practice to use ID
- security_group_id #security group for the instance (use the security group ID)
#network will be defined inside resources

parameters: #declares the parameters
  key_name:
    type: string
    default: <key-name>
    description: Key pair to use to be able to SSH into instance
  image_id:
    type: string
    default: <image-id> #Image ID
    description: The image for the instance will be IMAGE-NAME
  flavor_id:
    type: string
    default: <flavor-id> #Flavor ID
    description: The flavor for the instance will be FLAVOR-NAME
  security_group_id:
    type: string
    default: <security-group-id> #ID of the security group
    description: SECURITY-GROUP-NAME #this could be a default security group for example

resources: #declares the template resources
  test_instance: #name of the instance
    type: OS::Nova::Server
    properties:
      image: { get_param: image_id } #retrieves the image ID from image_id parameter
      flavor: { get_param: flavor_id } #retrieves the flavor ID from flavor_id parameter
      key_name: { get_param: key_name } #retrieves the key pair from key_name parameter
      networks:
        - network: Internal #define the internal network as Internal
      security_groups:
        - { get_param: security_group_id }

```

Using a template similar to this one, we can launch a stack.

### 3.8.3 Create a Stack

Stacks can be launched using the OpenStack CLI. The syntax for creating a stack is:

```

openstack stack create [-h] [-f {json,shell,table,value,yaml}]
                        [-c COLUMN] [-noindent] [-prefix PREFIX]
                        [-max-width <integer>] [-fit-width]
                        [-print-empty] [-e <environment>]
                        [-s <files-container>] [-timeout <timeout>]
                        [-pre-create <resource>] [-enable-rollback]
                        [-parameter <key=value>]
                        [-parameter-file <key=file>] [-wait]
                        [-poll SECONDS] [-tags <tag1,tag2...>]
                        [-dry-run] -t <template>
                        <stack-name>

```

For example, to create a stack using the template *example-template.yaml*:

```
openstack stack create -t example-template.yaml example-stack
```

This should return something similar to the following:

Field	Value
id	deda567a-4240-466d-9ac6-4bed4b848666
stack_name	example-stack
description	An example template which launches instances.
creation_time	2020-07-20T08:22:14Z
updated_time	None
stack_status	CREATE_IN_PROGRESS
stack_status_reason	Stack CREATE started

Then the status of the stack can be checked using the command:

```
openstack stack show <stack-id>
```

### 3.8.4 Delete a Stack

To delete a stack, use the command:

```
openstack stack delete <stack-id>
```

**Note:** Any resources such as instances which have been created specifically for the stack will also be deleted.

### 3.8.5 Further Reading

For more detailed information on Heat, and to see some example stacks, refer to the advance Heat docs at <https://stfc-cloud-docs.readthedocs.io/en/latest/Heat/index.html>

### 3.8.6 References

[https://docs.openstack.org/heat/train/template\\_guide/hot\\_guide.html](https://docs.openstack.org/heat/train/template_guide/hot_guide.html)

[https://docs.openstack.org/heat/train/template\\_guide/hot\\_spec.html#hot-spec](https://docs.openstack.org/heat/train/template_guide/hot_spec.html#hot-spec)

<https://www.cisco.com/c/dam/en/us/products/collateral/cloud-systems-management/metacloud/newbie-tutorial-heat.pdf>

## 3.9 Create a Virtual Machine from command line in STFC Openstack

It's common to want to create VMs from command line. To do this, you need to be on a host that has the openstack command line interface installed.

On our Scientific Linux 7 machines the correct repository is already set up so you can use you install the client by running:

```
yum install -y python-openstackclient
```

### 3.9.1 Setting up the environment to select project

After logging into a host, you have to set your environment variables.

You can download a script to set these up for you from: [https://openstack.stfc.ac.uk/project/api\\_access/openrc/](https://openstack.stfc.ac.uk/project/api_access/openrc/). You will need to be signed in to the project you wish to use in the OpenStack web interface for this to work.

Or set them yourself in a script:-

```
export OS_AUTH_URL=https://openstack.nubes.rl.ac.uk:5000/v3
export OS_IDENTITY_API_VERSION=3
export OS_PASSWORD=*****
export OS_PROJECT_DOMAIN_NAME=default
export OS_PROJECT_NAME="SCD Cloud Operations Team"
export OS_TENANT_NAME="SCD Cloud Operations Team"
export OS_USERNAME=<username>
export OS_USER_DOMAIN_NAME=<userdomain>
```

Take note of the “PROJECT-NAME” and “TENANT-NAME” – they are the project that you wish to create the new VMs in. The USER\_DOMAIN\_NAME and USERNAME are the user and the domain that user is in, that you are using to create the VMs. PASSWORD is your login to openstack – in clear text. It is common to put all of the above in a shell file (just as above, but with a file extension of .sh) and then “source” the environment variables so that they are part of your command shell So, if I put the above list of commands in a file “martin\_openstack.sh” then I need to run “source ./martin\_openstack.sh” after I have logged into a openstack server.

### 3.9.2 Finding images and flavors available

Once on the command line of a server, you need to list the flavors available for the project to use as well as the images available. To see the images available, run:-

```
openstack flavor list
```

You should expect an output of a table that looks something like the below:

Table 1: flavor

ID	Name	RAM	Disk	Ephemeral	VCPUs	Is Public
026ace2c-5247-4bdc-8929-81d129cc69bf	c3.small	4096	20	0	2	True
1	m1.tiny	1024	10	0	1	True
11	c1.xlarge	16384	100	0	4	True
110dd30b-3d5f-4a4a-b380-ba6dfcfa5ad3	c2.large	8192	80	0	8	True
12	c1.xxlarge	32768	160	0	8	True

continues on next page

Table 1 – continued from previous page

ID	Name	RAM	Disk	Ephemeral	VCPUs	Is Public
15bb0664-ffc8-45e4-9a99-06f3fcfcc680	m2.xlarge	16384	160	0	16	True
2	m1.small	2048	20	0	1	True
20c58ba0-f87c-423d-9ff1-2c7ae1bef7cc	c3.xlarge	32768	160	0	16	True
26a01e1d-2e32-4949-a390-d2e6383d2ae7	c1.3xl	49152	200	0	16	True
3	m1.medium	4096	40	0	1	True
34900a49-ad26-490d-95f5-87aeb6c12d7c	m3.xlarge	32768	160	0	16	True
37c83b1b-05ab-4169-9022-78b30f5450d8	m3.small	4096	20	0	2	True
4	m1.large	8192	80	0	2	True
45f004b0-97e9-403b-a63a-90ed9af70087	g-k620.xxlarge	32768	160	0	8	False
5	m1.xlarge	16384	160	0	8	True
52786a6d-e20a-4906-82db-4ce1d91a715f	m2.small	2048	20	0	2	True
6cf0813c-1ba2-4999-b7eb-34d71a2a4199	c3.medium	8192	40	0	4	True
6e6062f3-4744-4320-8dc3-767795ec98e8	m2.large	8192	80	0	8	True
6f5c0478-84c2-4478-a6d5-073d97a81c0e	c3.large	16384	80	0	8	True
78e477cc-a3e6-4ec8-af99-c59f33f02c3c	g-k620.tiny	1024	10	0	1	False
7ca6ba0f-416a-4a8c-87a7-0f2890c9fd11	m1.test	1024	10	0	1	False
7ff4bca9-2e13-42e0-9283-cf17cff372f3	c2.small	2048	20	0	2	True
8	c1.medium	4096	40	0	2	True
9	c1.large	8192	80	0	2	True
a033c03d-e684-47a7-be9f-a857de135c4c	c2.xlarge	16384	160	0	16	True
a7716bcf-490d-4c01-a518-b25587cc02e8	m3.large	16384	80	0	8	True
bcea5cd1-ccc1-45aa-a771-82cf2deb41ba	c2.medium	4096	40	0	4	True
c7ee6c89-3059-4bc1-b332-317bdc4da36	m3.medium	8192	40	0	4	True
ce0828cb-132c-4890-8b78-c7c123804e43	c1.4xl	92160	400	0	28	True
d0184b50-bce2-4679-9b00-c1b774f9c647	m3.tiny	2048	20	0	1	True
e166d59d-fab6-4839-9f04-ca4b275262c3	g-k620.4xl	128000	400	0	30	False
faa9265d-98e4-4cc6-acd7-fa8a7e72e8ef	m1.xxlarge	32768	160	0	8	False
fc04f5fc-c264-4aa9-b1bf-fc3aa7736cbc	m2.medium	4096	40	0	4	True

To see the choice of images available, run the command:-

```
openstack image list
```

You should expect an output of a table that looks something like the below:

Table 2: images

ID	Name	Status
b8c3c82e-1ba3-4c4e-9d09-eb713cbe52c6	Next3-ScientificLinux-7-Gui	active
d3becd76-8046-4c9e-ab9d-e476b40237c7	ScientificLinux-6-AQ	active
8ba8781a-87a9-4f11-ae57-3865c19e8be9	ScientificLinux-6-Gui	active
1bda5d33-b718-4a0e-a330-037e6096bb9c	ScientificLinux-6-NoGui	active
2e8fb278-c5d8-4647-b13c-e63c577fe4ae	ScientificLinux-7-AQ	active
44aa5e0e-cf74-4e71-ab2c-b11cf5dd1e66	ScientificLinux-7-Gui	active
3741c38f-f59a-4fd5-89b0-f61f2d577b23	ScientificLinux-7-NoGui	active
5d8dfe3b-52e0-48e1-9219-88c47dbd8c8a	Ubuntu-Bionic-Gui	active
02406ced-6980-4937-b9c5-38964cefd4d4	Ubuntu-Bionic-NoGui	active
f29f4278-f168-489d-ae54-7aa1269755f2	Ubuntu-Trusty-Gui	active
5a5178af-ef85-4184-bf4a-d607a43b248a	Ubuntu-Trusty-NoGui	active
24cde165-b797-4fce-8322-59cd36dc596a	Ubuntu-Xenial-Gui	active
e25b990f-8fd9-4a42-bf43-4d421f8e93e9	Ubuntu-Xenial-NoGui	active
190cda0b-ac8e-42a9-af49-38484c88ac63	readthedocs_snapshot_2018-10-25	active
147eefc8-ad2b-447f-8195-944fe4547ddd	xming_rdesktop_readthedocs_snapshot1	active

To see the list of networks available, run the command:-

```
openstack network list
```

...this returns two networks named “External” and “Internal”. Since we can’t add VMs directly to External network, we will be using the “Internal” network.

### 3.9.3 Putting it all together to create a new Instance

Here is an example command, putting together information from the previous commands:-

```
openstack server create --flavor m1.tiny --image ScientificLinux-7-NoGui --nic net-
↪id=Internal --security-group default --key-name xbe91637 test_2018-10-29_1511
```

...where flavor and image are from the previous commands used, net\_id is the name of the Network to be used (note you can use the actual Net\_ID number instead if preferred – it can make things faster!). Security group is defining the specific security group, and key-name, chooses the ssh keypair to include when creating the host. “test\_2018-10-29\_1511” is the name of the host that is being created – known within openstack. Some useful extras Adding –timing after the openstack command provides some statistics of how quickly various calls are being completed. You will see the usual host creation data, but at the end, you will also see the response times of each openstack API module.

```
openstack --timing server create --flavor m1.tiny --image Ubuntu-Xenial-NoGui --nic net-
↪id=Internal --security-group default --key-name xbe91637 test_2018-10-30_1357
```

To delete a host, you can use the command:-

```
openstack server delete <instance id>
```

You can also run with –debug after the openstack command – this will give you a step by step commentary as to what is happening when creating a virtual machine. For example:-

```
openstack --debug server create --flavor m1.tiny --image Ubuntu-Xenial-NoGui --nic net-
↪id=Internal --security-group default --key-name xbe91637 test_2018-10-30_1357
```

### 3.9.4 References

The following is a good generic guide:- <https://docs.openstack.org/mitaka/install-guide-ubuntu/launch-instance-provider.html>

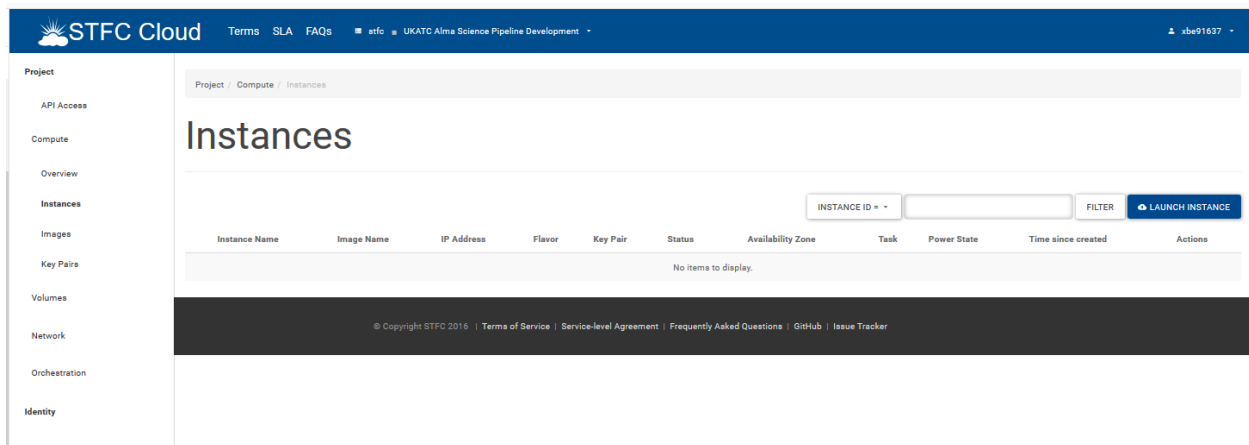
## 3.10 Create a Virtual Machine on Openstack using the Openstack Web interface

### 3.10.1 Scenario

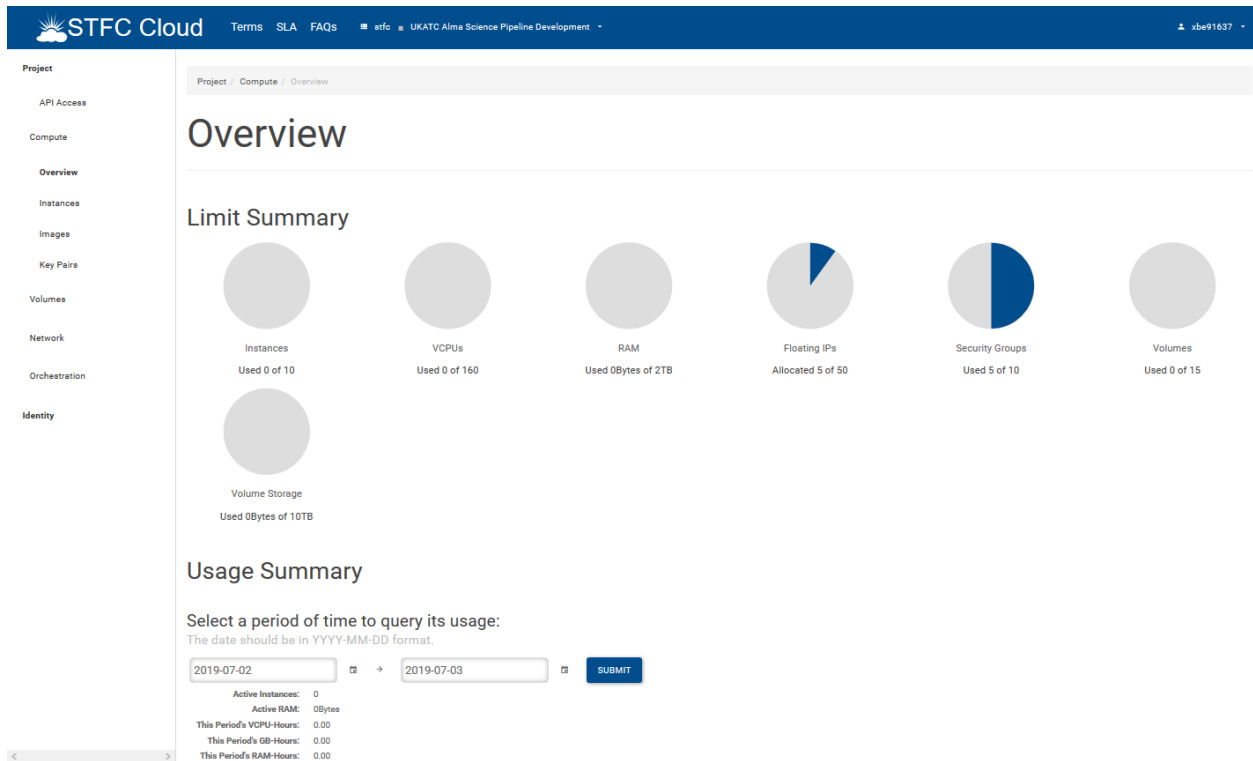
This document describes how to create a virtual machine once a project environment has been setup for a user. It assumes that the user has a login to openstack, and has already added their SSH public key into the openstack system.

### 3.10.2 Procedure

Login to the openstack interface. Select your project at the top of the screen if you have access to multiple projects.



Click on Compute, then Overview: This screen will show you the quota limits set up so far, and how much of those quota limits are consumed by your project. This is important as if you attempt to build a virtual machine that exceeds these limits, it will fail to build !



If the quotas look fine, and you do not need them expanding (should you need this, you should contact [cloud-support@stfc.ac.uk](mailto:cloud-support@stfc.ac.uk), after first checking with the other users of the project that all the VMs currently running or shut-down are needed !), Click on the Instances option on the Left Hand Side. Click on the “Launch Instance” button. You will see a pop-up window as follows:-

## Launch Instance



Details \*

Source \*

Flavor \*

Networks \*

Network Ports

Security Groups

Key Pair

Configuration

Server Groups

Scheduler Hints

Metadata

Please provide the initial hostname for the instance, the availability zone where it will be deployed, and the instance count. Increase the Count to create multiple instances with the same settings.

Instance Name \*

Description

Availability Zone

ceph

Count \*

1

Total Instances (10 Max)

10%

0 Current Usage

1 Added

9 Remaining

×

CANCEL

←

BACK

NEXT

→

LAUNCH INSTANCE

Some of the Left Hand Side menu options of the pop-up window have asterisks next to them (\*). This means that there is mandatory information that needs to be filled in, in order for Openstack to create the virtual machine. Click on “Details” first: Fill in an instance name in the field. This is how the machine is referred to within openstack. Fill in a description: This is more to help other users of the project, and cloud admins as to what the host is used for. Availability zone: This *should* be set to “Ceph”. If you have an option, select “Ceph”. (Ceph is the back-end storage system we use for Cloud hosts). Count: This must be set to at least “1”. You can create multiples of the same host at the same time, but creating larger numbers of hosts at the same can generate some build errors for some of the hosts (General rule of thumb is that generally, not more than 25 at a time). Now click on the “Source” option on the Left Hand Side:-

## Launch Instance



Details \*

Source \*

Flavor \*

Networks \*

Network Ports

Security Groups

Key Pair

Configuration

Server Groups

Scheduler Hints

Metadata

Instance source is the template used to create an instance. You can use an image, a snapshot of an instance (image snapshot), a volume or a volume snapshot (if enabled). You can also choose to use persistent storage by creating a new volume.



Select Boot Source

Image

Create New Volume

YES

NO

## Allocated

Name	Updated	Size	Type	Visibility
Select an item from Available items below				

v Available 10

Select one

Name	Updated	Size	Type	Visibility	
> Ubuntu-Bionic-NoGui	3/4/19 8:44 AM	10.00 GB	raw	Public	↑
> Ubuntu-Xenial-Gui	3/4/19 8:45 AM	10.00 GB	raw	Public	↑
> Ubuntu-Xenial-NoGui	3/4/19 8:45 AM	10.00 GB	raw	Public	↑
> Ubuntu-Trusty-Gui	2/27/19 11:01 AM	10.00 GB	raw	Public	↑
> Ubuntu-Bionic-Gui	2/27/19 4:23 PM	10.00 GB	raw	Public	↑
> Ubuntu-Trusty-NoGui	2/27/19 11:02 AM	10.00 GB	raw	Public	↑
> ScientificLinux-6-Gui	1/22/19 12:45 PM	10.00 GB	raw	Public	↑
> ScientificLinux-7-NoGui	2/21/19 3:33 PM	10.00 GB	raw	Public	↑
> ScientificLinux-6-NoGui	1/22/19 12:38 PM	10.00 GB	raw	Public	↑
> ScientificLinux-7-Gui	3/18/19 5:02 PM	10.00 GB	raw	Public	↑

x CANCEL

← BACK

NEXT →

LAUNCH INSTANCE

In the Select Boot Source, the default should be “Image” and “Create new volume” should be set to “no” unless you require your VM to have resilient storage for its boot drive. What you select in this field determines what you see as options in “Available” just below. “Image” should provide one of the standard template STFC Cloud ready images that we support: By default, we usually have the last 3 Ubuntu supported images, with a GUI and non-gui option, and we also have a Scientific Linux 6 and 7 gui and non-gui option. We are expecting to have a Centos 7 image available sometime in the near future. Select the image you wish by clicking on the up arrow the option in the Available area.

This should transfer it to the “Allocated” area. (We will not explore the other options for boot source in this scenario). Once an image is selected, click on the “Flavor” list in the Left Hand Side:-

Flavors manage the sizing for the compute, memory and storage capacity of the instance.

## Allocated

Select an item from Available items below

Available 28 Select one

Click here for filters. x

Name	VCPUS	RAM	Total Disk	Root Disk	Ephemeral Disk	Public
> c3.small	2	4 GB	20 GB	20 GB	0 GB	Yes
> m1.tiny	1	1 GB	10 GB	10 GB	0 GB	Yes
> c1.xlarge	4	16 GB	100 GB	100 GB	0 GB	Yes
> c2.large	8	8 GB	80 GB	80 GB	0 GB	Yes
> c1.xxlarge	8	32 GB	160 GB	160 GB	0 GB	Yes
> m2.xlarge	16	16 GB	160 GB	160 GB	0 GB	Yes
> m1.small	1	2 GB	20 GB	20 GB	0 GB	Yes
> c3.xlarge	16	32 GB	160 GB	160 GB	0 GB	Yes
> c1.3xl	16	48 GB	200 GB	200 GB	0 GB	Yes
> m1.medium	1	4 GB	40 GB	40 GB	0 GB	Yes
> m3.xlarge	16	32 GB	160 GB	160 GB	0 GB	Yes
> m3.small	2	4 GB	20 GB	20 GB	0 GB	Yes
> m1.large	2	8 GB	80 GB	80 GB	0 GB	Yes
> m1.xlarge	8	16 GB	160 GB	160 GB	0 GB	Yes

The Flavor determines the CPU, Memory and Disk combination. The list is much larger than the list shown above. It is suggested that you aim for CPU and Memory types that fit your needs best – it is possible to “resize” your virtual machine to a larger size later on, and also possible to add additional “volumes” to a Virtual machine if you need more disk space on it. Note the two different types “m” and “c”: The M types are shared CPU core, while C type are not shared (worthy of note that depending on which one you choose, it also determines which hypervisor type the Virtual machine will be scheduled to run on !). Click on the up arrow for the Flavor you wish to select. Click on “Networks” on the Left Hand Side:-

## Launch Instance



Details

Source

Flavor

Networks \*

Network Ports

Security Groups

Key Pair

Configuration

Server Groups

Scheduler Hints

Metadata

Networks provide the communication channels for instances in the cloud.



## ▼ Allocated

Select networks from those listed below.

Network	Subnets Associated	Shared	Admin State	Status
---------	--------------------	--------	-------------	--------

Select an item from Available items below

## ▼ Available 2

Select at least one network

Network	Subnets Associated	Shared	Admin State	Status	
> ukatc_aspd_private_net	ukatc_aspd_private_subnet	Yes	Up	Active	↑
> Internal	Internal4	Yes	Up	Active	↑
	Internal3				
	Internal1				
	Internal2				

× CANCEL

← BACK

NEXT →

LAUNCH INSTANCE

For Projects that just have one network type (Private or Internal) this is selected by default. A Hybrid project will have both a private network and an Internal Network, so one *must* be selected by clicking on an up arrow. That completes all of the mandatory Fields that you need to complete in order to create one or more virtual machines. At this point, the “Launch Instance” button should go dark blue, indicating that you can now create the VM. Other areas that you may wish to pay attention to when creating a VM would be “Security Groups” as different security groups are often needed depending on if the VM is on the Internal Network or a Private Network. You may also wish to select which SSH key you wish injected into the Virtual machine when it starts. The other options on the Left hand Side, you will probably not need for day to day use unless you have a specialist requirement. The system will then be seen in the instances screen at various stages of completeness:-

The screenshot shows the STFC Cloud web interface. The left sidebar contains navigation links: Project, API Access, Compute, Overview, Instances (selected), Images, Key Pairs, Volumes, Network, Orchestration, and Identity. The main content area is titled 'Instances' and shows a table with one instance named 'test'. The instance is in the 'Spawning' state, with a power state of 'No State' and a time since created of '0 minutes'. The table columns are: Instance Name, Image Name, IP Address, Flavor, Key Pair, Status, Availability Zone, Task, Power State, Time since created, and Actions. The 'test' instance has an image name of 'ScientificLinux-7-NoGui', IP address '172.16.114.108', flavor 'm1.tiny', key pair 'xbe91637', status 'Build', availability zone 'ceph', and task 'Spawning'. The 'Actions' column for the 'test' instance has a button labeled 'ASSOCIATE FLOATING IP'.

You may see “Scheduling” and “Building” in the Task column as well. Once it is ready to connect to, you should see something like:-

The screenshot shows the STFC Cloud web interface. The left sidebar contains navigation links: Project, API Access, Compute, Overview, Instances (selected), Images, Key Pairs, Volumes, Network, Orchestration, and Identity. The main content area is titled 'Instances' and shows a table with one instance named 'test'. The instance is in the 'Active' state, with a power state of 'Running' and a time since created of '0 minutes'. The table columns are: Instance Name, Image Name, IP Address, Flavor, Key Pair, Status, Availability Zone, Task, Power State, Time since created, and Actions. The 'test' instance has an image name of 'ScientificLinux-7-NoGui', IP address '172.16.114.108', flavor 'm1.tiny', key pair 'xbe91637', status 'Active', availability zone 'ceph', and task 'None'. The 'Actions' column for the 'test' instance has a button labeled 'CREATE SNAPSHOT'.

At this stage, you may not be able to SSH onto it quite yet, but you may be able to log into the console, or at least see how far it has got in the boot process.

### 3.10.3 Common Things to do After Creating your VM

-You may wish to SSH to it: This you can only do if it was created on the “Internal” network, otherwise, you will have to assign a floating IP address to the VM so that you can SSH into it. -You may wish to login using the console: This will only work if you are using a STFC FEDID as your Openstack user account. If you are not, then you need to SSH onto the VM first, create a user account, password, home directory, and entry in the /etc/sudoers.d/cloud file for the new user – this will then allow that user to login to the console. -You may wish to update the host (yum update, or apt-get update; apt-get upgrade). You may find that you cannot run “yum” the first time you login: This is because all Scientific Linux 6 and Scientific Linux 7 Virtual machines are partially managed by the configuration management tool in order to keep them up to date and well prepared. It is suggested that after the first 10 minutes or so, you reboot the VM so that all the updates will take effect. -A hybrid project will not see the “ScientificLinux7-AQ” or “ScientificLinux6-AQ” images by default: These are the fully Aquilon managed VMs and are available only to STFC folks who wish to use the Aquilon configuration management tool. Note that this can only be done on these image types, and if the hosts are on the “Internal” network.

## 3.11 Deleting Virtual Machine

Deleting entities is the only way to **release resources** and **free up quotas**.

**Warning:** You should always refer to *Creating Snapshots from Instance* as this process is **not reversible** and may result in **data loss**.

### 3.11.1 Web Interface

1. In Web Interface *Compute* → *Instances*, select the VM you wish to delete

## Instances

INSTANCE ID =

FILTER

LAUNCH INSTANCE

DELETE INSTANCES

MORE ACTIONS

Displaying 7 items

<input type="checkbox"/>	Instance Name	Image Name	IP Address	Flavour	Key Pair	Status	Availability Zone	Task	Power State	Age	Actions
<input checked="" type="checkbox"/>	delete-cli	ubuntu-focal-20.04-nogui	172.16.113.107	c3.small	-	Active	ceph	None	Running	1 minute	<div>CREATE SNAPSHOT</div> <div></div>

2. Click *DELETE INSTANCES*

## Confirm Delete Instances



▲ **Warning:** Deleted instances are not recoverable.

You have selected:

- "delete-web"

CANCEL

DELETE INSTANCES

### 3.11.2 Command-line

See *Using OpenStack Command-line Interface* on how to set-up the command line client.

1. Find the instance ID with `openstack server list`

```
$ openstack server list
```

ID	Name	Status	Networks
c3b01eac-b5c7-4f57-8b09-f575f37cfe25	delete-cli	ACTIVE	Internal=172.16.113.107
ubuntu-focal-20.04-nogui			c3.small

2. Run

```
openstack server delete <instance-id>
```

Example

```
$ openstack server delete c3b01eac-b5c7-4f57-8b09-f575f37cfe25
```

## 3.12 Grow the disk of an Openstack VM by cloning the machine

### Table of Contents

- *Grow the disk of an Openstack VM by cloning the machine*
  - *Scenario*
  - *Method*
    - \* *WebUI*
    - \* *Command-Line*
  - *Confirm the storage*

### 3.12.1 Scenario

In the STFC cloud, if you created a new VM in instances, the default setting in the *Sources* was to create the VM with a image. If you run out of disk space on the VM, you can usually *resize* it to a new Flavor, and you will have the same VM, but with more space.

**Note:** See *Resize a VM* on how to resize a VM.

You can also create a new instance with *Volume* as *Sources*. However, with *Volumes*, the system ignores the disk sizing aspect of the *flavor*. This document describes how to create a clone of the original VM and increase the disk size, so that space is available on the new VM.

### 3.12.2 Method

#### WebUI

**Note:** Check your Quotas for our project: Volume sizes, number of volumes, and number of volume snapshots are all parameters for which your project has a quota for, and there is a fair chance that what you have allocated may not be enough. Check with a cloud admin first before proceeding if you are unsure.

1. Go to *Compute* → *Instances* in the LHS menu
2. Click the drop-down menu on the right-hand side (in *Actions* column) and select *SHUT OFF INSTANCE*

## Instances

INSTANCE ID =

FILTER

LAUNCH INSTANCE

DELETE INSTANCES

MORE ACTIONS ▾

Displaying 17 items

<input type="checkbox"/>	Instance Name	Image Name	IP Address	Flavour	Key Pair	Status	Availability Zone	Task	Power State	Age	Actions
<input type="checkbox"/>	volume launch 2	ubuntu-focal-20.04-nogui	172.16.101.212	c3.small	-	Active	ceph	None	Running	22 minutes	<div>CREATE SNAPSHOT ▾</div> <div> ASSOCIATE FLOATING IP  ATTACH INTERFACE  DETACH INTERFACE  EDIT INSTANCE  ATTACH VOLUME  DETACH VOLUME  UPDATE METADATA  EDIT SECURITY GROUPS  EDIT PORT SECURITY GROUPS  CONSOLE  VIEW LOG  RESCUE INSTANCE  PAUSE INSTANCE  SUSPEND INSTANCE  SHELVE INSTANCE  RESIZE INSTANCE  LOCK INSTANCE  SOFT REBOOT INSTANCE  HARD REBOOT INSTANCE  SHUT OFF INSTANCE </div>

3. Click on the instance name

# Instances

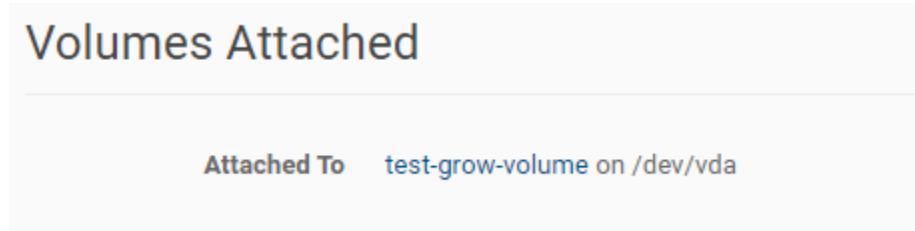
INSTANCE ID = ▾

FILTER

Displaying 17 items

<input type="checkbox"/>	Instance Name	Image Name	IP Address	Flavour	Key Pair	Status	Availability Zone
<input type="checkbox"/>	volume launch 2	ubuntu-focal-20.04-nogui	172.16.101.212	c3.small	-	Active	ceph

4. Check the *Volumes Attached* for the volume attached to */dev/vda* and click on the volume name



5. Open the drop-down menu on the top left corner and click *CREATE SNAPSHOT*

Project / Volumes / Volumes / test-grow-volume

# test-grow-volume

EDIT VOLUME ▾

MANAGE ATTACHMENTS  
CREATE SNAPSHOT  
CHANGE VOLUME TYPE  
UPLOAD TO IMAGE  
UPDATE METADATA

Overview

Snapshots

Name

test-grow-volume

6. Give it a name and click *CREATE VOLUME SNAPSHOT (FORCE)*

# Create Volume Snapshot

This volume is currently attached to an instance. In some cases, creating a snapshot from an attached volume can result in a corrupted snapshot.

Snapshot Name

Description

## Description:

From here you can create a snapshot of a volume.

## Snapshot Limits

Total Gibibytes

334 of 3,000 GiB Used

Number of Snapshots

3 of 50 Used

CANCEL

CREATE VOLUME SNAPSHOT (FORCE)

**Note:** Once the volume is attached to a VM, you cannot remove it: the only way to remove the VM host from the volume is if the instance is created, and when the VM was created, the setting for the volume to persist still exists.

- Go to *Volume* → *Volumes* in the LHS menu, then click on + *CREATE VOLUME*

## Volumes

Filter

+ CREATE VOLUME

ACCEPT TRANSFER

DELETE VOLUMES

- Give it a name and in *Volume Source* Select *SNAPSHOT*; In *Use snapshot as a source* select the snapshot you created and input the *Size (GiB)* you need. After that, Click *CREATE VOLUME*

×

## Create Volume

**Volume Name**

**Description**

**Volume Source**

SNAPSHOT

Use snapshot as a source <sup>\*</sup>

TEST-GROW-IMAGE (20 GiB)

Size (GiB) <sup>\*</sup>

30

-

+

**Group** ⓘ

NO GROUP

### Description:

Volumes are block devices that can be attached to instances.

### Volume Limits

**Total Gibibytes** 334 of 3,000 GiB Used

**Number of Volumes** 13 of 100 Used

CANCEL

CREATE VOLUME

9. Refer to *Create a Virtual Machine on Openstack using the Openstack Web interface* except in *Source* you should select *Select Boot Source* → *Volume* and select the volume you created from the snapshot

---

**Note:** The flavor of the new VM does not affect the disk size – only the volume parameters does this !

---

## Launch Instance



Details \*

Source \*

Flavour \*

Networks \*

Network Ports

Security Groups

Key Pair

Configuration

Server Groups

Scheduler Hints

Metadata

Price

Instance source is the template used to create an instance. You can use an image, a snapshot of an instance (image snapshot), a volume or a volume snapshot (if enabled). You can also choose to use persistent storage by creating a new volume.

Select Boot Source

Volume

Delete Volume on Instance Delete

YES

NO

## Allocated

Name	Description	Size	Type	Availability Zone
Select an item from Available items below				

## Available 2

Select one



Click here for filters or full text search.



Name	Description	Size	Type	Availability Zone
> test-grow		30 GB	qcow2	▲ ceph
>				▲ ceph

30 GB

qcow2

▲ ceph



&gt;

▲ ceph



X CANCEL

← BACK

NEXT →

LAUNCH INSTANCE

**Note:** When creating a new instance from a volume, the volume may still have previous users created in the system and their home directories and they may have their details still in the `/etc/sudoers.d/cloud` file. However, you will not find that your *FEDID* has been added to the `/etc/sudoers.d/cloud` file.

## Command-Line

1. Get the ID for the Instance you want to grow with `openstack server list`.

```
$ openstack server list
```

ID	Name	Status	Networks
	Image		Flavor

(continues on next page)

(continued from previous page)

```

↩-----+-----+-----+-----+-----+-----+-----+-----+
| 2f2c54ac-bae0-4f26-a4bb-7e9ff429f3b4 | volume launch 2          | ACTIVE | ↩
↩Internal=172.16.101.212 |                               | c3.
↩small      |
+-----+-----+-----+-----+-----+-----+-----+-----+
↩-----+-----+-----+-----+-----+-----+-----+-----+

```

2. Get the details of the VM with `openstack server show <Instance-ID>` Note down the ID of `volumes_attached`.

```

$ openstack server show 2f2c54ac-bae0-4f26-a4bb-7e9ff429f3b4
+-----+-----+-----+-----+-----+-----+-----+-----+
↩-----+
| Field                                | Value                                | ↩
↩
+-----+-----+-----+-----+-----+-----+-----+-----+
↩-----+
| OS-DCF:diskConfig                    | AUTO                                | ↩
↩
| OS-EXT-AZ:availability_zone          | ceph                                | ↩
↩
| OS-EXT-STS:power_state                | Running                             | ↩
↩
| OS-EXT-STS:task_state                 | None                                | ↩
↩
| OS-EXT-STS:vm_state                   | active                              | ↩
↩
| OS-SRV-USG:launched_at                | 2022-02-04T08:24:54.000000         | ↩
↩
| OS-SRV-USG:terminated_at              | None                                | ↩
↩
| accessIPv4                            |                                     | ↩
↩
| accessIPv6                            |                                     | ↩
↩
| addresses                             | Internal=172.16.101.212            | ↩
↩
| config_drive                          | True                                | ↩
↩
| created                               | 2022-02-04T08:24:31Z               | ↩
↩
| flavor                               | c3.small (026ace2c-5247-4bdc-8929-81d129cc69bf) | ↩
↩
| hostId                               | 206ef5cda6ec5d9c47a3fed0c7cf1980c98cf4b11e870195380699ca ↩
↩
| id                                    | 2f2c54ac-bae0-4f26-a4bb-7e9ff429f3b4 | ↩
↩
| image                                 |                                     | ↩
↩
| key_name                              | None                                | ↩
↩
| name                                  | volume launch 2                     | ↩

```

(continues on next page)

(continued from previous page)

```

↪      |
| progress                | 0
↪      |
| project_id              | 80ab2bd11e5f46bf96bf47658d07499d
↪      |
| properties              |
↪      |
| security_groups         | name='default'
↪      |
| status                  | ACTIVE
↪      |
| updated                 | 2022-02-04T08:24:54Z
↪      |
| user_id                 |
↪ 3ae4ecf4b9e0e66260b7aaebc2cc98aac3c95221e42f1cb49113ed751d8b9f2c |
| volumes_attached       | id='d7f3c63e-4d01-4959-bde7-fe55032243fd'
↪      |
+-----+
↪-----+

```

3. Shutdown the VM using `openstack server stop <Instance-ID>`

```
$ openstack server stop 2f2c54ac-bae0-4f26-a4bb-7e9ff429f3b4
```

4. Create a snapshot using `openstack volume snapshot create --volume --force <snapshot-name>`

```

$ openstack volume snapshot create --volume d7f3c63e-4d01-4959-bde7-fe55032243fd --force
↪ test-cli-grow-snapshot
+-----+
| Field      | Value
+-----+
| created_at | 2022-02-04T13:44:49.884301
| description | None
| id          | 2e09549b-b02c-494f-a704-0a882d0ef96c
| name        | test-cli-grow-snapshot
| properties  |
| size        | 30
| status      | creating
| updated_at  | None
| volume_id   | d7f3c63e-4d01-4959-bde7-fe55032243fd
+-----+

```

5. Create a volume using the snapshot `openstack volume create --snapshot <snapshot-id> --size <size-in-GiB> <name>`

```

$ openstack volume create --snapshot 2e09549b-b02c-494f-a704-0a882d0ef96c --size 35 test-
↪ grow-cli-volume
+-----+
↪ +
| Field      | Value
↪ |
+-----+
↪ +

```

(continues on next page)

(continued from previous page)

```

| attachments          | []
↪ |
| availability_zone     | ceph
↪ |
| bootable              | true
↪ |
| consistencygroup_id  | None
↪ |
| created_at            | 2022-02-04T13:48:24.000000
↪ |
| description           | None
↪ |
| encrypted             | False
↪ |
| id                    | 37c8b414-f2c9-452c-885b-8cffc185c596
↪ |
| multiattach           | False
↪ |
| name                  | test-grow-cli-volume
↪ |
| properties            |
↪ |
| replication_status    | None
↪ |
| size                  | 35
↪ |
| snapshot_id           | 2e09549b-b02c-494f-a704-0a882d0ef96c
↪ |
| source_volid          | None
↪ |
| status                | creating
↪ |
| type                  | __DEFAULT__
↪ |
| updated_at            | None
↪ |
| user_id               | 3ae4ecf4b9e0e66260b7aaebc2cc98aac3c95221e42f1cb49113ed751d8b9f2c
↪ |
+-----+-----+
↪ +

```

6. Create a VM using the volume `openstack server create --volume <volume-id> --flavor <flavor> --network <network> <name>`

```

$ openstack server create --volume 37c8b414-f2c9-452c-885b-8cffc185c596 --flavor c3.
↪ small --network Internal test-cli-grow-vm
+-----+-----+
↪ -----+
| Field                  | Value
↪      |
+-----+-----+
↪ -----+

```

(continues on next page)

(continued from previous page)

OS-DCF:diskConfig	MANUAL	
↪		
OS-EXT-AZ:availability_zone		
↪		
OS-EXT-STS:power_state	NOSTATE	
↪		
OS-EXT-STS:task_state	scheduling	
↪		
OS-EXT-STS:vm_state	building	
↪		
OS-SRV-USG:launched_at	None	
↪		
OS-SRV-USG:terminated_at	None	
↪		
accessIPv4		
↪		
accessIPv6		
↪		
addresses		
↪		
adminPass	wXJMroU7AhVZ	
↪		
config_drive		
↪		
created	2022-02-04T13:50:31Z	
↪		
flavor	c3.small (026ace2c-5247-4bdc-8929-81d129cc69bf)	
↪		
hostId		
↪		
id	f3d5f09a-74f1-42ea-8823-a45429aa0eb9	
↪		
image		
↪		
key_name	None	
↪		
name	test-cli-grow-vm	
↪		
progress	0	
↪		
project_id	80ab2bd11e5f46bf96bf47658d07499d	
↪		
properties		
↪		
security_groups	name='default'	
↪		
status	BUILD	
↪		
updated	2022-02-04T13:50:31Z	
↪		
user_id		
↪ 3ae4ecf4b9e0e66260b7aaebc2cc98aac3c95221e42f1cb49113ed751d8b9f2c		

(continues on next page)

(continued from previous page)

```
| volumes_attached |
↪ |
+-----+
↪ -----+
```

### 3.12.3 Confirm the storage

You can confirm the size of the disk using `lsblk`

```
$ lsblk
NAME        MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
loop0       7:0      0 61.9M  1 loop /snap/core20/1242
loop1       7:1      0 55.5M  1 loop /snap/core18/2253
loop2       7:2      0 73.1M  1 loop /snap/lxd/21902
loop3       7:3      0 55.5M  1 loop /snap/core18/2284
loop4       7:4      0 32.5M  1 loop /snap/snapd/13640
loop5       7:5      0 43.4M  1 loop /snap/snapd/14549
loop6       7:6      0 76.2M  1 loop /snap/lxd/22340
loop7       7:7      0 61.9M  1 loop /snap/core20/1328
sr0         11:0     1 470K   0 rom  /mnt/context
vda         252:0     0   35G   0 disk
├─vda1      252:1     0  34.9G  0 part /
├─vda14     252:14    0    4M   0 part
└─vda15     252:15    0  106M  0 part /boot/efi
```

## 3.13 Manually update Scientific Linux and Centos Virtual Machines

When using our provided Scientific Linux or Centos images you may occasionally find that you get an error that a repository cannot be found when installing a package.

This is because we use snapshotted repositories for many of the common software repositories in order to minimise the risk of incompatible or broken packages. We periodically validate these snapshots and release them to our users and delete snapshots which are over 1 year old.

Our configuration management system periodically updates the configuration of these within your VMs.

If this hasn't happened it may be necessary to run these components manually. This process also maintains the monitoring and access for the Cloud Team

You can do that with the following command:

```
quattor-fetch && quattor-configure --all
```

You should then see an output ending with this line:

```
[OK]    0 errors, 0 warnings executing configure
```

Warning are usually fine and transient but if there are any errors or if this doesn't resolve your initial issue then please contact the Cloud team.

## 3.14 Prevent Automatic Updates

It is also possible to prevent our curated images from automatically updating certain packages or completely.

### 3.14.1 Preventing Automatic Updates of some packages

It is possible to version lock certain packages you care about by changing the owner of `/etc/yum/pluginconf.d/versionlock.list`:

```
sudo chown your-username /etc/yum/pluginconf.d/versionlock.list
```

Then add any packages to the above file in the same format as the other entries.

### 3.14.2 Preventing Automatic Updates completely

This is not recommended and must only be done with the express written consent of the Cloud team. This can be obtained by emailing [cloud-support@stfc.ac.uk](mailto:cloud-support@stfc.ac.uk)

You still must comply with our terms of service.

Auto updating can be disabled by creating a file at `/etc/noquattor`. This file should contain the date the file is created and who created it.

## 3.15 Rebuild a VM with a new image

**Warning:** Rebuilding Instances will result in loss of data. You should always refer to *Creating Snapshots from Instance* to backup your data.

OpenStack has a useful rebuild function that allows you to rebuild an instance from a fresh image while maintaining the same fixed and floating IP addresses, amongst other metadata.

### 3.15.1 Web Interface

1. Log-in to the STFC cloud (<https://openstack.stfc.ac.uk/>)
2. In the Web Interface, Go to *Compute* → *Instances*
3. Click the drop-down menu on the right-hand side (in *Actions* column) and select *REBUILD INSTANCE*

# Instances

INSTANCE ID =  FILTER [LAUNCH INSTANCE](#) [DELETE INSTANCES](#) [MORE ACTIONS](#)

Displaying 16 items

<input type="checkbox"/>	Instance Name	Image Name	IP Address	Flavour	Key Pair	Status	Availability Zone	Task	Power State	Age	Actions
<input type="checkbox"/>	test-rebuild	ubuntu-20.04-no gui	172.16.101.195	c3.small		Active	ceph	None	Running	1 minute	<a href="#">CREATE SNAPSHOT</a> <a href="#">ASSOCIATE FLOATING IP</a> <a href="#">ATTACH INTERFACE</a> <a href="#">DETACH INTERFACE</a> <a href="#">EDIT INSTANCE</a> <a href="#">ATTACH VOLUME</a> <a href="#">DETACH VOLUME</a> <a href="#">UPDATE METADATA</a> <a href="#">EDIT SECURITY GROUPS</a> <a href="#">EDIT PORT SECURITY GROUPS</a> <a href="#">CONSOLE</a> <a href="#">VIEW LOG</a> <a href="#">RESCUE INSTANCE</a> <a href="#">PAUSE INSTANCE</a> <a href="#">SUSPEND INSTANCE</a> <a href="#">SHELVE INSTANCE</a> <a href="#">RESIZE INSTANCE</a> <a href="#">LOCK INSTANCE</a> <a href="#">SOFT REBOOT INSTANCE</a> <a href="#">HARD REBOOT INSTANCE</a> <a href="#">SHUT OFF INSTANCE</a> <a href="#">REBUILD INSTANCE</a>

4. Select the image from the drop-down menu in the pop-up. Click on the *REBUILD* button.

## Rebuild Instance



Select Image \*

SCIENTIFICLINUX-7-NOGUI (20.0 GB) ▾

Disk Partition

AUTOMATIC ▾

Description

### Description:

Select the image to rebuild your instance.

CANCEL

REBUILD INSTANCE

- The instance will be rebuilt with the new image

## Instances

INSTANCE ID ▾

FILTER

LAUNCH INSTANCE

DELETE INSTANCES

MORE ACTIONS ▾

Displaying 16 items

<input type="checkbox"/>	Instance Name	Image Name	IP Address	Flavour	Key Pair	Status	Availability Zone	Task	Power State	Age	Actions
<input type="checkbox"/>	test-rebuild	scientificlinux-7-nogui	172.16.101.195	c3.small		Active	ceph	None	Running	12 minutes	CREATE SNAPSHOT ▾

### 3.15.2 Command-line

**Note:** See *Using OpenStack Command-line Interface* on how to set-up the command line client.

- Get the Server ID (The VM to rebuild) and the Image ID

```
$ openstack image list
+-----+-----+
| ID                                     | Name                                     |
+-----+-----+
| ae4a8e18-627c-484a-9909-7bd93d9e3e97 | ubuntu-focal-20.04-gui                 |
| active                               |                                         |
+-----+-----+
```

(continues on next page)

(continued from previous page)

```

+-----+
↪ +-----+
$ openstack server list
+-----+
↪ +-----+
↪ +-----+
| ID                                     | Name                               | Status | Networks |
↪                                     | Image                             |        |          |
↪ Flavor                               |                                   |        |          |
+-----+
↪ +-----+
↪ +-----+
| b5acb398-76b4-48fe-9b9a-d480636fd9 | test-rebuild                     | ACTIVE |          |
↪ Internal=172.16.101.195             | scientificlinux-7-nogui          |        |          |
↪                                     | c3.small                         |        |          |
+-----+
↪ +-----+
↪ +-----+

```

## 2. Run

```
openstack server rebuild --image <image-id> <server-id>
```

### Example

```

$ openstack server rebuild --image ae4a8e18-627c-484a-9909-7bd93d9e3e97 b5acb398-76b4-
↪ 48fe-9b9a-d480636fd9
+-----+
| Field          | Value                               |
+-----+
| OS-DCF:diskConfig | MANUAL                             |
| accessIPv4       |                                     |
| accessIPv6       |                                     |
| addresses        | Internal=172.16.101.195            |
| adminPass        | yoTq5HDrhH6a                      |
| created          | 2021-12-03T10:31:27Z               |
| flavor           | c3.small (026ace2c-5247-4bdc-8929-81d129cc69bf) |
| hostId           | ff3c5830640867370bd8fb9228356baee63ff24e5baa381e41798dc9 |
| id               | b5acb398-76b4-48fe-9b9a-d480636fd9 |
| image            | ubuntu-focal-20.04-gui (ae4a8e18-627c-484a-9909-7bd93d9e3e97) |
| name             | test-rebuild                       |
| progress         | 0                                   |
| project_id       | 80ab2bd11e5f46bf96bf47658d07499d |
| properties       |                                     |
| status           | REBUILD                            |
| updated          | 2021-12-03T10:57:41Z               |
| user_id          | 3ae4ecf4b9e0e66260b7aaebc2cc98aac3c95221e42f1cb49113ed751d8b9f2c |
+-----+

```



# Resize Instance

**Flavour Choice** \*

Advanced Options

Old Flavour

c3.small

New Flavour \* ?

C3.MEDIUM

## Flavour Details

Name	c3.medium
VCPUs	4
Root Disk	40 GB
Ephemeral Disk	0 GB
Total Disk	40 GB
RAM	8,192 MB

## Project Limits

Number of Instances of Used

Number of VCPUs of Used

Total RAM of MB Used

CANCEL

RESIZE

3.16.2 Command-line

**Note:** See *Using OpenStack Command-line Interface* on how to set-up the command line client.

- 1. Get the serverID (the VM to resize) and flavorID

```
$ openstack server list
+-----+-----+-----+-----+
| ID                                     | Name                               | Status | Networks |
| Flavor                                | Image                             |        |           |
+-----+-----+-----+-----+
| b5acb398-76b4-48fe-9b9a-d480636fd9 | test-rebuild                     | SHUTOFF |           |
Internal=172.16.101.195                | ubuntu-focal-20.04-gui          |        |           |
| c3.small                             |                                  |        |           |
+-----+-----+-----+-----+

$ openstack flavor list
+-----+-----+-----+-----+-----+-----+
| ID                                     | Name                               | RAM | Disk | Ephemeral |
| VCPUs | Is Public |                                     |                                     |                                     |
+-----+-----+-----+-----+-----+-----+
| 6cf0813c-1ba2-4999-b7eb-34d71a2a4199 | c3.medium                         | 8192 | 40 | 0 |
4 | True                                |                                     |                                     |                                     |
+-----+-----+-----+-----+-----+-----+
```

- 2. Run

```
openstack server resize --flavor <flavor-id> <server-id>
```

Example

```
$ openstack server resize --flavor 6cf0813c-1ba2-4999-b7eb-34d71a2a4199 b5acb398-76b4-
48fe-9b9a-d480636fd9
```

## 3.17 Running an NGinX webserver inside a Docker container

This tutorial shows how to run a webserver with docker.

Firstly create a VM - the instructions here are based on our ubuntu-focal-20.04-nogui image.

SSH to your VM and install docker:

```
sudo apt-get update && sudo apt-get install docker.io
```

Start and enable docker

```
sudo systemctl enable docker && sudo systemctl start docker
```

Add current user to the docker group (**Please do NOT run docker and containers as root**)

```
sudo usermod -aG docker ${USER}
```

Or if you need to add a user that is not logged in

```
sudo usermod -aG docker username
```

To apply new group you must logout and log back in

```
exit  
ssh user@<your_ip>
```

You can confirm changes have applied by running

```
groups  
#Output:  
wheel docker
```

Pull the docker image for the nginx webserver

```
docker pull nginx
```

Run the nginx docker container:

```
docker run -it --rm -d -p 80:80 --name nginx nginx
```

Check that the container is running

```
docker ps
```

You should see the nginx container running

You should now be able to browse to your webserver at the <http://<instance-ip>>

If this doesnt work check the security group for your instance, enabling http traffic through.

## 3.18 Share a directory between Linux hosts on the same Network using NFS

### 3.18.1 Scenario

You have created a linux Scientific Linux 7 host on a Private (i.e 192.168.x.y ) network, which has a volume attached to it, and the volume has been “partitioned” and “formatted” with a linux file system and “mounted” on /data. This document describes how you can share that volume with other linux hosts on the same private network using Linux’s NFS server service. Important Security Note: For security reasons, this tech note is not appropriate for setting up a NFS server via a Float IP address (i.e 130.246.x.y address) or on the “internal network” (the host has a 172.16.x.y IP address). This is due to security reasons: The file share could easily be exposed to outside of your project without care for the Security Groups applied.)

### 3.18.2 Setting up the NFS server

On the NFS Server:

1. Set the correct permissions for the file-share.
2. Un-mount the volume on /data using the command “umount /data”.
3. Run the command “sudo chown nobody:nobody /data”
4. Run the command “chmod 777 /data” This changes the ownership and the permissions of the fileshare to that any user on the remote system will be able to read and write data to it.

```
drwxrwxrwx 2 nobody nobody 6 Jul 10 15:00 data
```

5. Now re-mount the volume using the command:-

```
mount /dev/vdb1 /data
```

6. Check it is mounted using the command:-

```
df -h
```

7. Check the output, the line /dev/vdb1 is what we are looking for. If you have not already, make sure the following line is in the /etc/fstab file so that the volume mounts on host boot (otherwise you may be sharing out an empty directory !):-

```
[root@test-nfs /]# df -h
Filesystem      Size  Used Avail Use% Mounted on
/dev/uda1       20G   5.0G   16G   25% /
devtmpfs        1.9G     0   1.9G    0% /dev
tmpfs           1.9G     0   1.9G    0% /dev/shm
tmpfs           1.9G   8.5M   1.9G    1% /run
tmpfs           1.9G     0   1.9G    0% /sys/fs/cgroup
tmpfs          379M     0   379M    0% /run/user/0
/dev/sr0        458K   458K     0  100% /mnt/context
/dev/vdb1       19G   1.1G   17G    7% /data
[root@test-nfs /]#
```

Specifically this line

```
/dev/vdb1 /data ext4 defaults 0 0
```

8. Now create the configuration file that tells the NFS server which area of the filesystem to share out, and what the restrictions and parameters are.

```
sudo echo "/data 192.168.248.0/24(rw,sync,no_subtree_check)" >> /etc/exports
```

Where:

- /data is the area to share out,
- 192.168.248.0/24 is the subnet range of IP addresses that are allowed to connect to the file share.
- The parameters in brackets : rw means read and write, sync means that the data has to be written to disk before applying it,
- no\_subtree\_check prevents checking of directory trees.

9. You need to check that the NFS-server.service is enabled and running:

```
sudo systemctl list-unit-files --type=service
```

System should respond with something like:-

nfs-blkmap.service	disabled
nfs-config.service	static
nfs-idmap.service	static
nfs-idmapd.service	static
nfs-lock.service	static
nfs-mountd.service	static
nfs-rquotad.service	disabled
nfs-secure.service	static
nfs-server.service	disabled
nfs-utils.service	static
nfs.service	disabled
nfslock.service	static

10. In this case, we need to enable the nfs-server.service, and make sure it starts up when the system reboots. Run the command:

```
systemctl enable nfs-server.service
```

11. Run the following command to export the filesystem in /default:-

```
sudo exportfs
sudo exportfs -a
```

12. The system should show all the filesystems being “exported” by the NFS server in the /etc/exports file:-

```
[root@test-nfs ~]# exportfs
/data 192.168.248.0/24
[root@test-nfs ~]#
```

At this stage – the server side is now setup.

## Setting up the Security Group in Openstack

As the NFS server and NFS clients are on a “private” 192.168 network (in this example, they are on a 192.168.248.0/24 network), a security group should be applied that lets the NFS clients talk to the NFS server.

In this example, we are going to have a fairly open Security group so that other hosts within this specific Private Openstack project can see each other for all services. You can setup more details security groups for NFS – see the references for “NFS and security”. In the Horizon Openview Web Gui, navigate on the Left Hand Side menus to Network-> Security Groups

The screenshot shows the STFC Cloud Horizon Openview Web GUI. The left sidebar contains a navigation menu with options like Project, API Access, Compute, Volumes, Network, Network Topology, Networks, Routers, Security Groups, Floating IPs, Orchestration, Admin, and Identity. The main content area is titled 'Security Groups' and shows a table of existing security groups. At the top right of the table, there are buttons for '+ CREATE SECURITY GROUP' and 'DELETE SECURITY GROUPS'.

Name	Security Group ID	Description	Actions
Allow_all_local	3b2aaa0-7922-4a11-a75e-b05766b383c	Allow local private subnet to talk to other VMs locally	MANAGE RULES
check_DNS	a978eb27-9bd2-4b9e-a322-f32ba98a3fc8	make sure can access DNS locally	MANAGE RULES
default	099f2a1f-f59f-4db1-b0ef-37a021cd1af8	Default security group	MANAGE RULES
demo	e932517e-cae5-4cae-8ba9-897e02b8862d		MANAGE RULES
webserver	1ab60c78-7570-4cb5-9369-60db881a5bf7	Opens ports 80 and 443 for a webserver, also includes ports from default security group	MANAGE RULES

Click on “+Create Security Group” Give the Security groups a name and a description, then click on the “Create Security Group”:-

The screenshot shows the 'Create Security Group' dialog box. It has a title bar with a close button (X). The main content area has two sections: 'Name' and 'Description'. The 'Name' field contains the text 'Allow\_all\_local'. The 'Description' field contains the text 'allow any port inbound between local subnet hosts.' To the right of the description field, there is a paragraph of text: 'Security groups are sets of IP filter rules that are applied to network interfaces of a VM. After the security group is created, you can add rules to the security group.' At the bottom of the dialog, there are two buttons: 'CANCEL' and 'CREATE SECURITY GROUP'.

Create a rulebase that looks as follows – allowing “all ports” inbound for UDP and TCP:-

Project / Network / Security Groups / Manage Security Group Rule...

## Manage Security Group Rules: Allow\_all\_local (3b2aaad0-7922-4a11-a75e-bf05766b383c)

Displaying 2 items

+ ADD RULE DELETE RULES

<input type="checkbox"/> Direction	Ether Type	IP Protocol	Port Range	Remote IP Prefix	Remote Security Group	Actions
<input type="checkbox"/> Ingress	IPv4	TCP	1 - 65535	192.168.248.0/24	-	DELETE RULE
<input type="checkbox"/> Ingress	IPv4	UDP	1 - 65535	192.168.248.0/24	-	DELETE RULE

Displaying 2 items

© Copyright STFC 2016 | Terms of Service | Service-level Agreement | Frequently Asked Questions | GitHub | Issue Tracker

This will allow the other local hosts on your project private network full port access to each other. Add this new security group to your NFS server host, and any NFS client hosts. You can do this by navigating to the “Instances” window on the Left Hand Side Menu, then clicking on the down arrow in the Actions column of the VM host you wish to add the security group to, then select “Edit Security groups”. You can then add the security group you created to the VM:-

## Edit Instance

Information \* Security Groups

Add and remove security groups to this instance from the list of available security groups.

All Security Groups

Filter

c_rally_bd2892d1_...	+
c_rally_673e37c6_...	+
c_rally_9bef30b7_...	+

Instance Security Groups

Filter

default	-
Allow_all_local	-

### 3.18.3 Setting up the NFS Clients

Ensure that you have added the security group to the NFS client host. Make sure the directory for where you wish to mount the remote file-system exists: In this case /data. You can create it with the command:-

```
sudo mkdir /data
```

You can “manually” mount the NFS share using the following command:-

```
sudo mount 192.168.248.26:/data /data
```

...where the IP address 192.168.248.26 is the Private IP address of the NFS server, and the /data is the directory that is being exported. The second “/data” is the “mount point” of where the remote file system is mounted. You can check if the filesystem is mounted by issuing the command:-

```
df -h
```

which should show something like:- [root@testing-day16-round1-24 ~]# df -h

Filesystem	Size	Used	Avail	Use%	
Mounted on /dev/vda1	10G	9.2G	862M	92%	
/dev/tmpfs	487M	0	487M	0%	
/dev/shm	496M	0	496M	0%	
/run/tmpfs	496M	51M	446M	11%	
/sys/fs/cgroup	tmpfs	100M	0	100M	0%
/run/user/0	/dev/sr0	458K	458K	0	100%
/mnt/context	192.168.248.26:/data	19G	1.1G	17G	7%

If you wish to mount the remote NFS share when the NFS client machine boots, you can add the following line to the /etc/fstab file:-

```
192.168.248.26:/data /data nfs defaults
```

All users on the NFS clients should now be able to read and write to the /data directory. Further things you may wish to investigate and make use of It is possible to create “quotas” on the filesystem that is exported via NFS, so that you can account for how much disk space each user and group are using. You can enforce these quotas such that a particular user account can’t use all the disk space up. In this model, it is possible that more than one user will have the same userID, so they will have access to each others files on the file-system. It is possible to create a centralised userID server that can make each using have a unique userid when accessing the VMs and the shared Volume (Setting up a NIS sever with Home NFS directories would be an example of this). You can export multiple filesystems from different disks at the same time. You can test the speed of writing a file from the NFS client to the NFS server using a command such as:-

```
time dd -if=/dev/zero of=1GB_test.iso bs=1M count=1024
```

will return something like 1024+0 records in 1024+0 records out 1073741824 bytes (1.1 GB) copied, 6.77108 s, 159 MB/s

real 0m6.804s user 0m0.001s sys 0m0.664s ...so 1Gbyte was saved in 6.77 seconds – 159Mbytes per second.

### 3.18.4 References

<https://vitux.com/install-nfs-server-and-client-on-ubuntu/>  
<https://www.tecmint.com/add-disk-larger-than-2tb-to-an-existing-linux/> NFS security: <http://tldp.org/HOWTO/NFS-HOWTO/security.html>

## 3.19 Create Instance Snapshot and importing snapshot as image

A snapshot can be used as a **backup** or **template** for creating new instances.

### Table of Contents

- *Create Instance Snapshot and importing snapshot as image*
  - *Creating Snapshots from Instance*
    - \* *Web Interface*
    - \* *Command-Line*
  - *Downloading snapshot*
  - *Create new image from snapshot*
    - \* *Import snapshot to project*
    - \* *Booting From Image*
  - *Example*

### 3.19.1 Creating Snapshots from Instance

#### Web Interface

1. Go to *Compute* → *Instance* in Web Interface (<https://openstack.stfc.ac.uk/>)
2. Shut off your instance by clicking the drop-down menu on the right-hand side (in *Actions* column) and select *SHUT OFF INSTANCE*

# Instances

INSTANCE ID =  FILTER **LAUNCH INSTANCE** **DELETE INSTANCES** MORE ACTIONS ▾

Displaying 16 items

<input type="checkbox"/>	Instance Name	Image Name	IP Address	Flavour	Key Pair	Status	Availability Zone	Task	Power State	Age	Actions
<input type="checkbox"/>	test-rebuild	ubuntu-focal-20.04-gui	172.16.101.195	c3.small	DonaldUserKeyPair	Active	ceph	None	Running	58 minutes	CREATE SNAPSHOT ASSOCIATE FLOATING IP 3 days ATTACH INTERFACE 23 hours DETACH INTERFACE EDIT INSTANCE 1 week ATTACH VOLUME 6 days DETACH VOLUME UPDATE METADATA 2 weeks EDIT SECURITY GROUPS 2 weeks EDIT PORT SECURITY GROUPS CONSOLE 2 weeks VIEW LOG 1 day RESCUE INSTANCE PAUSE INSTANCE 2 weeks SUSPEND INSTANCE 1 day SHELVE INSTANCE RESIZE INSTANCE 2 weeks LOCK INSTANCE 1 day SOFT REBOOT INSTANCE 2 weeks HARD REBOOT INSTANCE 1 day SHUT OFF INSTANCE

3. Create a Snapshot by clicking the drop-down menu on the right-hand side (in *Actions* column) and select *CREATE SNAPSHOT*

## Instances

INSTANCE ID =  FILTER **LAUNCH INSTANCE** **DELETE INSTANCES** MORE ACTIONS ▾

Displaying 16 items

<input type="checkbox"/>	Instance Name	Image Name	IP Address	Flavour	Key Pair	Status	Availability Zone	Task	Power State	Age	Actions
<input type="checkbox"/>	test-rebuild	ubuntu-focal-20.04-gui	172.16.101.195	c3.small		Shutoff	ceph	None	Shut Down	1 hour, 53 minutes	START INSTANCE
<input type="checkbox"/>	test-rebuild	ubuntu-focal-20.04-gui	172.16.101.195	c3.small		Active	ceph	None	Running	2 days, 23 hours	CREATE SNAPSHOT

4. Give it a name and click *CREATE SNAPSHOT*

# Create Snapshot



Snapshot Name \*

## Description:

A snapshot is an image which preserves the disk state of a running instance.

CANCEL

CREATE SNAPSHOT

## Command-Line

**Note:** See [Using OpenStack Command-line Interface](#) on how to set-up the command line client.

1. Find the server ID (Instance) using the command below

```
$ openstack server list
+-----+-----+-----+-----+
| ID                                     | Name               | Status | Networks |
| Flavor                                | Image              |        |           |
+-----+-----+-----+-----+
| b5acb398-76b4-48fe-9b9a-d480636fd9 | test-rebuild       | ACTIVE | 172.16.101.195 |
| c3.small                            | ubuntu-focal-20.04-gui |        |               |
+-----+-----+-----+-----+
```

2. Shut down the instance using

```
$ openstack server stop <server-id>

#example
$ openstack server stop b5acb398-76b4-48fe-9b9a-d480636fd9
```

3. Confirm that the server is shut-off

```
$ openstack server list
+-----+-----+-----+-----+
```

(continues on next page)

(continued from previous page)

ID	Name	Status	Networks
Flavor	Image		
b5acb398-76b4-48fe-9b9a-d480636fd9	test-rebuild	SHUTOFF	
Internal=172.16.101.195	ubuntu-focal-20.04-gui		
c3.small			

#### 4. Use `openstack server image create` to create a snapshot

```
$ openstack server image create --name test-snapshot b5acb398-76b4-48fe-9b9a-d480636fd9
```

Field	Value
created_at	2021-12-03T12:37:40Z
file	/v2/images/2b9c6711-4dd8-4e5c-9edc-dd106b8319b5/file
id	2b9c6711-4dd8-4e5c-9edc-dd106b8319b5
min_disk	20

(continues on next page)

(continued from previous page)

```

↳
| min_ram      | 0
↳
↳
↳
↳
| name         | test-snapshot
↳
↳
↳
↳
| owner        | fa0f417fb4b5462791e4320e317eb2d2
↳
↳
↳
↳
| properties   | base_image_ref='90e1b77b-4192-46f1-8d9c-49fc36d9b54c', boot_roles='user',
↳ clean_attempts='2', description='Ubuntu-Focal-Gui', image_location='snapshot', image_
↳ state='available', image_type='snapshot', instance_uuid='b5acb398-76b4-48fe-9b9a-
↳ d480636fd9', locations='[]', os_distro='Ubuntu', os_hidden='False', os_variant='Gui',
↳ os_version='20.04-Focal' |
| protected    | False
↳
↳
↳
↳
| schema       | /v2/schemas/image
↳
↳
↳
↳
| status       | queued
↳
↳
↳
↳
| tags         |
↳
↳
↳
↳
| updated_at   | 2021-12-03T12:37:40Z
↳
↳
↳
↳
| visibility   | private
↳
↳
↳
↳
+-----+

```

(continues on next page)

(continued from previous page)

```

↪ -----+
↪ -----+
↪ -----+
↪ -----+

```

#### 5. Check the image list

```

openstack image list
+-----+-----+
↪ -----+-----+
| ID                               | Name                               |
↪      | Status                    |                                     |
+-----+-----+-----+
↪ -----+-----+-----+
| 2b9c6711-4dd8-4e5c-9edc-dd106b8319b5 | test-snapshot                     |
↪      | active                        |                                     |
+-----+-----+-----+
↪ -----+-----+-----+

```

### 3.19.2 Downloading snapshot

#### 1. Check the image ID of the snapshot

```

$ openstack image list
+-----+-----+
↪ -----+-----+
| ID                               | Name                               |
↪      | Status                    |                                     |
+-----+-----+-----+
↪ -----+-----+-----+
| 2b9c6711-4dd8-4e5c-9edc-dd106b8319b5 | test-snapshot                     |
↪      | active                        |                                     |
+-----+-----+-----+
↪ -----+-----+-----+

```

#### 2. Run

```
openstack image save --file <file-name> <image-id>
```

*#example*

```
$ openstack image save --file snapshot.raw 0258526c-f523-4645-8a9d-f6980ad87864
```

### 3.19.3 Create new image from snapshot

#### Import snapshot to project

**Note:** See *Using OpenStack Command-line Interface* on how to set-up the command line client.

You must *Downloading snapshot* to local computer first.

1. Run

```
openstack image create --container-format bare --disk-format qcow2 --file <path-to-image-
↳ file> <name>
```

Example

```
$ openstack image create --container-format bare --disk-format qcow2 --file snapshot.raw
↳ test-snapshot
```

#### Booting From Image

Create a VM and selecting the image as the boot source

#### Example

1. Find the image ID

```
$ openstack image list
+-----+-----+
↳ -----+-----+
| ID                                     | Name                                     |
↳      | Status      |
+-----+-----+
↳ -----+-----+
| 2b9c6711-4dd8-4e5c-9edc-dd106b8319b5 | test-snapshot                         |
↳      | active      |
+-----+-----+
↳ -----+-----+
```

2. Create the VM

```
$ openstack server create --flavor c3.small --image 2b9c6711-4dd8-4e5c-9edc-dd106b8319b5
↳ new-instance-from-snapshot
```

## 3.20 Use Cloud-init to configure a VM to meet the STFC Cloud Terms of Service

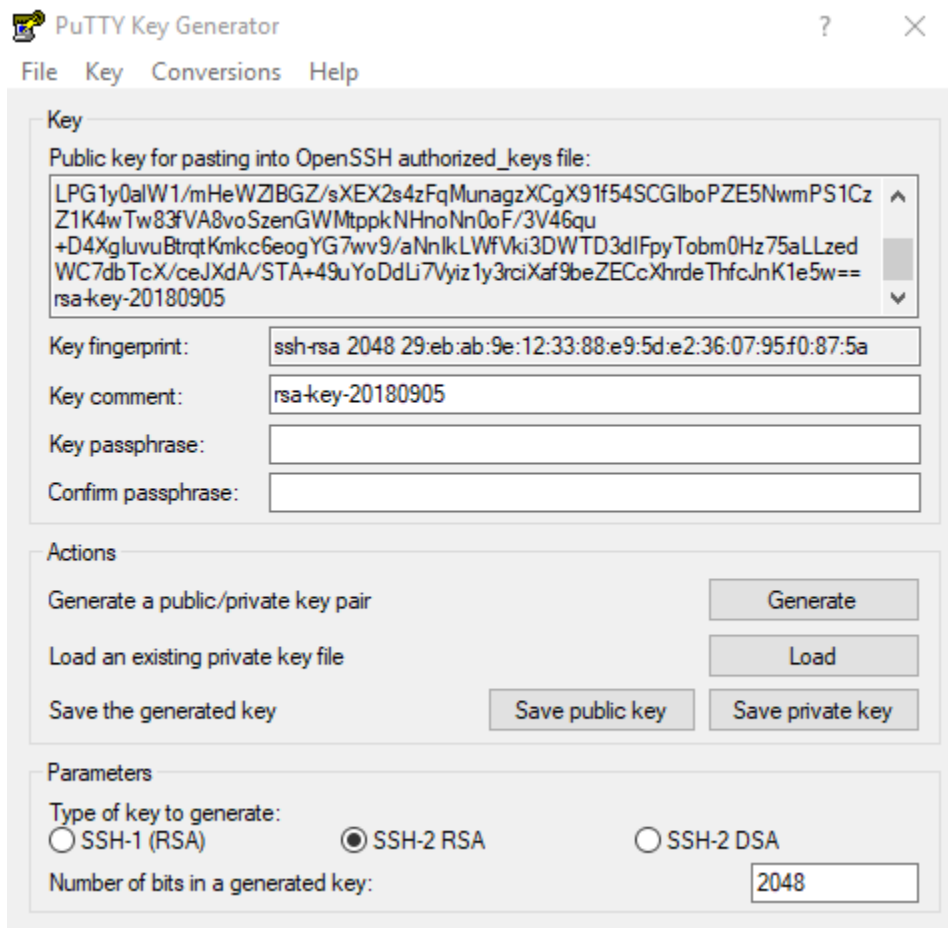
Documentations coming soon

## 3.21 Using Putty and Filezilla to connect to an Openstack Virtual machines

Most users often want to use a command line prompt, and be able to transfer files to and from their desktop machine to the virtual Linux host they are using. Tools like Putty and Filezilla are commonly used for this purpose.

For Putty and for Filezilla to connect to an Openstack Virtual Machine, you need to generate a “key pair” - which you do with PuTTYgen. This will create a “public” and “Private” key.

The default settings should be fine (ssh-2 RSA) and 2048 length. Longer lengths are more secure.



Save the public key - you need to copy this key and paste it into the <https://new-cloud.stfc.ac.uk> . Login as usual to this Web interface, then click on your login id in the Top Right Hand Side corner:-

My Machines

Martin Summers

Current Project: SCD Cloud Operati

SSH Key

Logout

Search:

Created	Type	Flavor	CPU	RAM
a day ago	ScientificLinux-7-AQ	m1.tiny	1	1GB
a day ago	ScientificLinux-7-NoGui	m1.tiny	1	1GB

Select “SSH-key”. Paste your public key into the field displayed :-

## SSH Key

You can have your SSH public key automatically added to new machines to allow passwordless login. Just paste your *public* key in the box below.

Key Name: xbe91637

SSH Key

```
ssh-rsa
AAAAB3NzaC1yc2EAAAQBApgeVJVdmem4lbJtBk6MIKpkiwHYGFYIKcH3n76Qd6fmLCJJ4IAIkE9RbqAdP1wUPgGqAfD+71j6Rt3CaZRPC95qODgEsnaJYM5HPE
WAC0+edCEZmKfXJllaMgsFubxtPxxXmsb/PmsbU/UcEQMxIBkMrqJ/qYJ81wcZJf1cSD4TsYhiS6UpdKHuSRX/aOdM1aWEIKTQqVNLqFcmPlAQxcaAmUxkEfCQ1mk3jig8lXYkaQ
X0HYF5FO5QYMc67r1IzPtl0vGJxEL5/T3LcNgwZB3d4B/DOFjWkx+hx5gLfw2SVLhZNtt7c3B+wyldT9uBYzaKwDjJw6w==
```

Submit

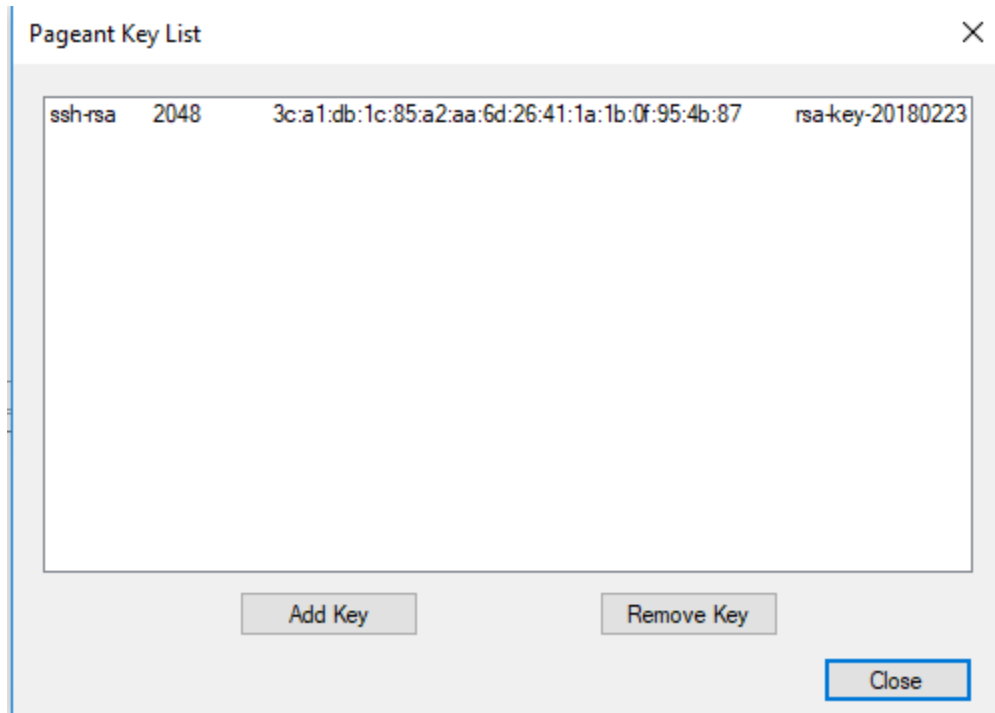
Click on the Submit key.

In Putty Key Generator – Enter a Key passphrase – this is the password you need to unlock access to your private key. (Do not make this empty – it must have a password for security reasons!). Once filled in, click on the “save private key”. Save this file somewhere safe – and make a copy of it somewhere safe (like a USB stick). Take note of your password in whatever method you use to store passwords safely. (You now have an authentication system which comprises of “something you have” – your private key, and “something you know”, your password.

You can now use Putty to authenticate to any new VM’s that you create:-

- 1) Download “pageant.exe” – it is a putty ssh agent and can be used to store your private key.
- 2) Run pageant.exe – it will appear as an icon in your tool bar:-



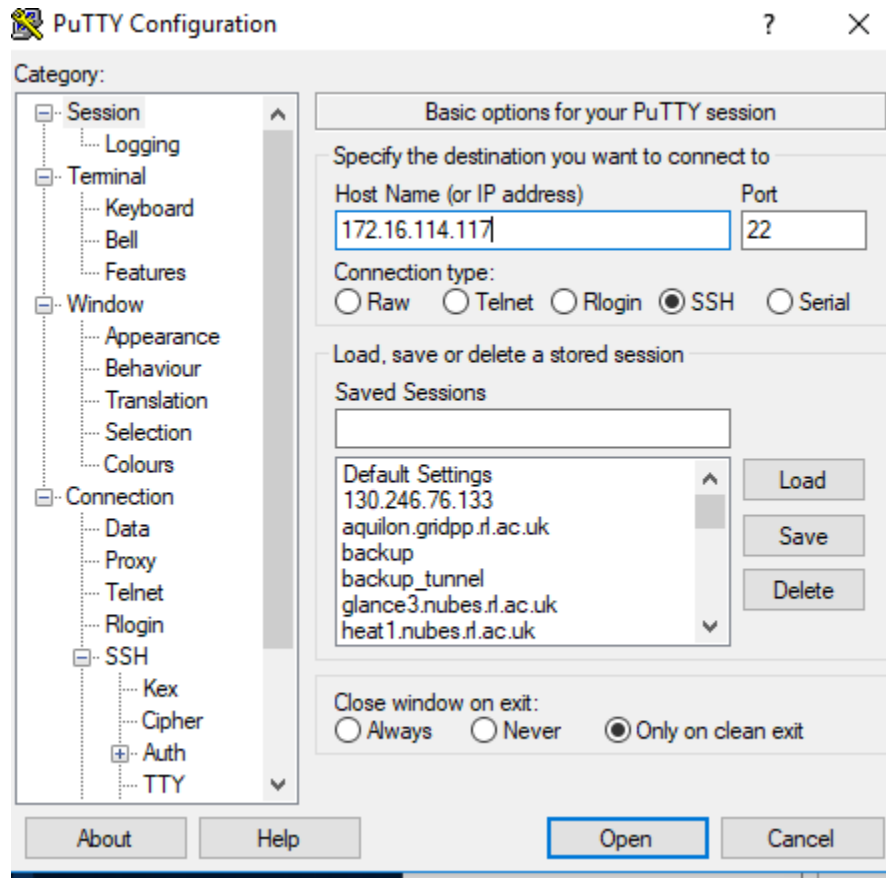


(click on the arrow) Right click on the icon in the toolbar (after clicking on the arrow), and select “add key”:-

Browse to where your private key is located – USB stick or local disk. You will be prompted for a password: This is the password to your private key.

Now start putty in the usual way:-

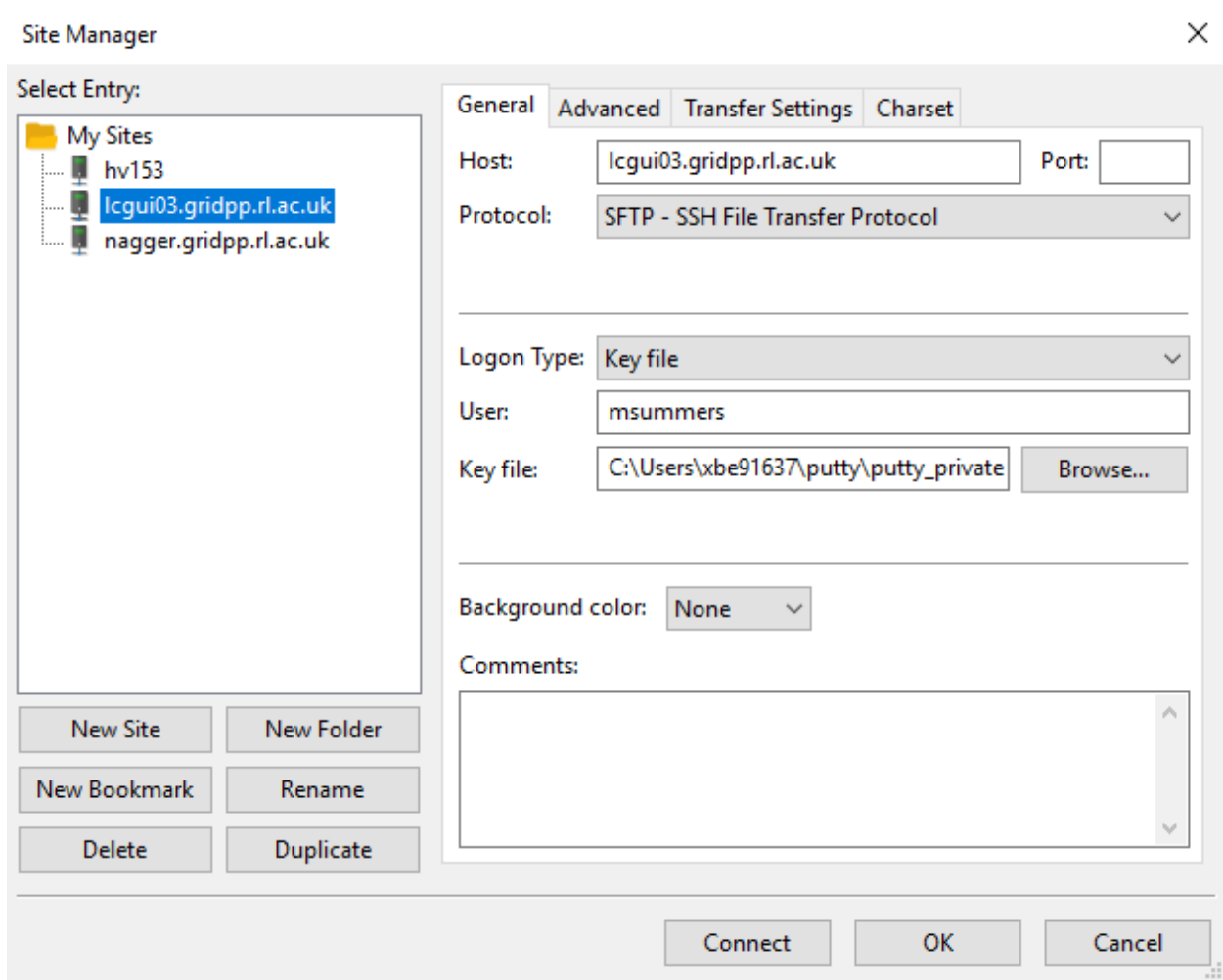
Enter the IP address of the virtual machine in the Hostname field:-



Click “Open” – you will be prompted for a username – (your username) - the system will not ask for a password as it will use the ssh-agent we added your private key to earlier. At this stage, you should be logged in!

Note that if you created a Virtual machine in Openstack before adding in your public key, you can add it afterwards from the console. This can be installed manually by pasting the key into the `/home/<user name>/.ssh/authorized_keys` file, where `<user name>` is your login name that you logged into the host with.

Filezilla (a tool for transferring files to and from a remote host) can be made to work in a similar way (i.e using your private key). Startup Filezilla, select File, then Site manager.



Click on “new site” – fill in the details as per the example above, where Host= IP address of the virtual machine, Protocol = SFTP, Logon type = “Key file”, username –your username and the location of your private key file. Save the site details and then press “connect”: you should be prompted for a password of your private key. Once entered, you should be connected.

Note that in all of these examples, I have assumed that you can directly connect to the virtual machine. If you are using an “IPSEC based VPN” solution, then it should still all work.

## 3.22 How to use IAM Credentials on the command line

To use IAM credentials on the command line you first need to generate an application credential.

To be able to create application credentials an additional role needs to be added to your user in OpenStack. This is done automatically once you have had a VM running for at least 1 hour.

To create an application credential, log in to OpenStack. Expand Identity and click on Application Credentials as shown below

Instances - OpenStack Dashboard

openstack.stfc.ac.uk/project/instances/

STFC Cloud

Terms SLA FAQ default wjc16017 wjc16017-admin

Project / Compute / Instances

# Instances

INSTANCE ID = FILTER LAUNCH INSTANCE DELETE INSTANCES MORE ACTIONS

Displaying 12 Items

Instance Name	Image Name	IP Address	Flavour	Key Pair	Status	Availability Zone	Task	Power State	Age	Actions
centos 8 ks	centos 8-stre am-8-nogui	172.16.114.81	c2.small	wjc16017	Active	ceph	None	Running	1 month, 1 week	CREATE SNAPSHOT
alextest	scientificlinux-7-a	172.16.103.161	c3.small	-	Active	ceph	None	Running	1 month, 1 week	CREATE SNAPSHOT

https://openstack.stfc.ac.uk/identity/application\_credentials/

Then Click on create application credentials

Application Credentials - OpenStack Dashboard

openstack.stfc.ac.uk/identity/application\_credentials/

STFC Cloud

Terms SLA FAQ default wjc16017 wjc16017-admin

Identity / Application Credentials

# Application Credentials

Filter + CREATE APPLICATION CREDENTIAL

Name	Project ID	Description	Expiration	ID	Roles	Actions
No items to display.						

© Copyright STFC 2016 | Terms of Service | Service-level Agreement | Frequently Asked Questions | GitHub | Issue Tracker

Fill in the details here as required. We recommend setting the expiration date of the credential for no more than 1 month from the date you create the credential (if you have need of longer credentials contact the cloud team).

If you need to make use of heat or create magnum clusters then tick the Unrestricted box. This allows the application credential to create a delegated user scope which is used by heat and magnum to perform actions on the user's behalf. Then hit Create Application Credential.

The screenshot shows a web browser window with the URL `openstack.stfc.ac.uk/identity/application_credentials/`. A modal dialog titled "Create Application Credential" is open. The dialog has a sidebar on the left with a navigation menu containing "Project", "Identity", "Projects", and "Application Credentials". The main content area of the dialog contains the following fields and instructions:

- Name:** A text input field containing "test-for-docs".
- Description:** A large text area for description.
- Secret:** A text input field for the secret.
- Expiration Date:** A date picker showing "31/12/2021".
- Expiration Time:** A time picker showing "--:--".
- Roles:** A dropdown menu with "user" selected.
- Unrestricted (dangerous):** An unchecked checkbox.

On the right side of the dialog, there is a "Description:" section with the following text:

Create a new application credential.

The application credential will be created for the currently selected project.

You may provide your own secret, or one will be generated for you. Once your application credential is created, the secret will be revealed once. If you lose the secret, you will have to generate a new application credential.

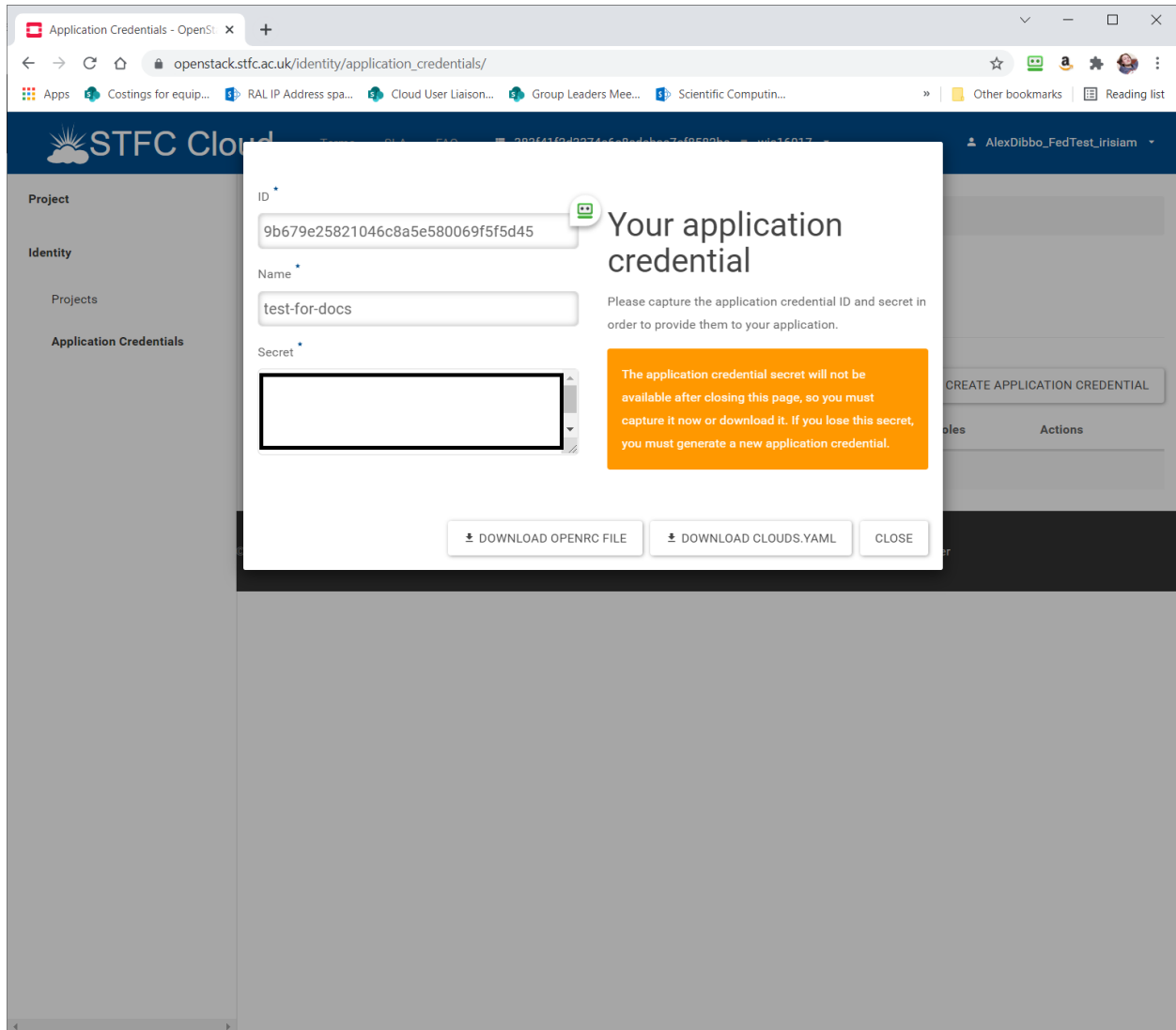
You may give the application credential an expiration. The expiration will be in UTC. If you provide an expiration date with no expiration time, the time will be assumed to be 00:00:00. If you provide an expiration time with no expiration date, the date will be assumed to be today.

You may select one or more roles for this application credential. If you do not select any, all of the roles you have assigned on the current project will be applied to the application credential.

By default, for security reasons, application credentials are forbidden from being used for creating additional application credentials or keystone trusts. If your application credential needs to be able to perform these actions, check "unrestricted".

At the bottom of the dialog are two buttons: "CANCEL" and "CREATE APPLICATION CREDENTIAL".

After a little thinking time you should see the following screen (secret redacted in the screen shot). From here hit Download Openrc File.



The file should look something like this:

```
#!/usr/bin/env bash

export OS_AUTH_TYPE=v3applicationcredential
export OS_AUTH_URL=https://openstack.stfc.ac.uk:5000/v3
export OS_IDENTITY_API_VERSION=3
export OS_REGION_NAME="RegionOne"
export OS_INTERFACE=public
export OS_APPLICATION_CREDENTIAL_ID=9b679e25821046c8a5e580069f5f5d45
export OS_APPLICATION_CREDENTIAL_SECRET=<redacted>
```

Where the credential ID and Secret match what is shown in the Your application credential screen.

You should then be able to source the openrc file and run openstack commands as below:

```
source Downloads/app-cred-test-for-docs-openrc.sh
openstack server list
```

## 3.23 Using OpenStack Command-line Interface

This tutorial is based on Ubuntu 20.04. At the end of the tutorial, you will have a machine that can control your cloud project using OpenStack CLI.

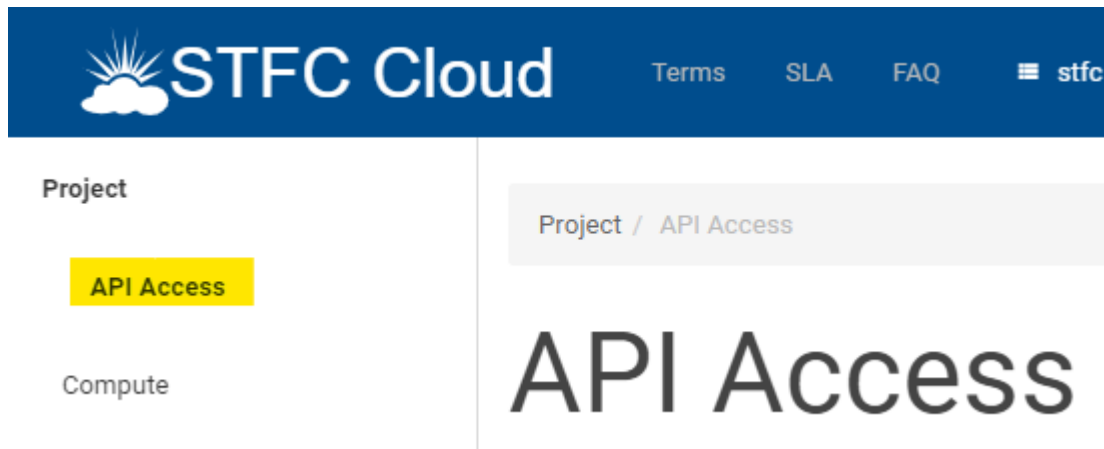
### 3.23.1 Install client

1. Install python, pip and openstackclient

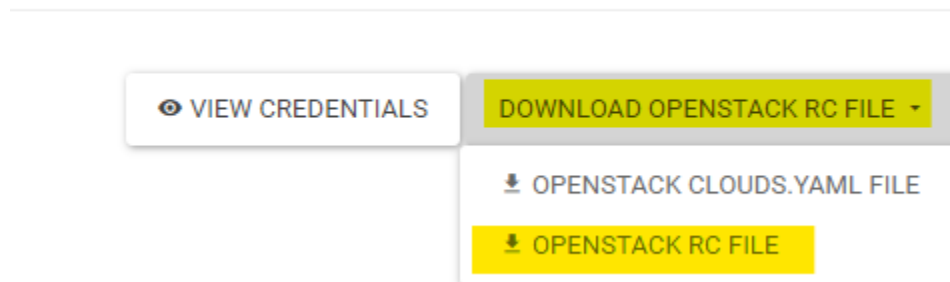
```
sudo apt install python-dev python3-pip python3-openstackclient
```

### 3.23.2 Log-in with Openstack CLI

2. in *Project* → *API Access*



3. Press *DOWNLOAD OPENSTACK RC FILE* drop-down
4. Press *Openstack RC FILE* to download the rc file.



5. Run

```
source <path-to-openstack-rc-file>
```

### 3.23.3 Verify connection

6. Run `openstack server list` and check if you can see the instances

```
$ openstack server list
+-----+-----+-----+-----+
↪-----+
↪-----+
| ID                                     | Name                               | Status | Networks |
↪                                     | Image                             |        |           |
↪ Flavor                               |                                   |        |           |
+-----+-----+-----+-----+
↪-----+
↪-----+
| 6b2bedc4-9d8e-4bf3-be63-1dd49bc2e188 | test-resize-rebuild              | ACTIVE |          |
↪ Internal=172.16.102.207                | ubuntu-focal-20.04-gui          |        |           |
↪                                     | c3.small                        |        |           |
+-----+-----+-----+-----+
↪-----+
↪-----+
```

## 3.24 Volumes in OpenStack

A volume is a **detachable block storage device**, similar to a USB hard drive. You can attach a volume to only **one** instance. Use the openstack client commands or the Web Interface to create and manage volumes.

You can also snapshot a volume to act as a backup or template for creating new volumes.

### Table of Contents

- *Volumes in OpenStack*
  - *Create Volume*
    - \* *Web Interface*
    - \* *Command-Line*
      - *Create simple volume*
      - *Create image from snapshot/image*
  - *Attaching Volumes to an instance*
    - \* *Web Interface*
    - \* *Command-Line*
  - *Accessing the volume*
  - *Detaching Volume*
    - \* *Web Interface*
    - \* *Command-line*
  - *Create Volume Snapshot*

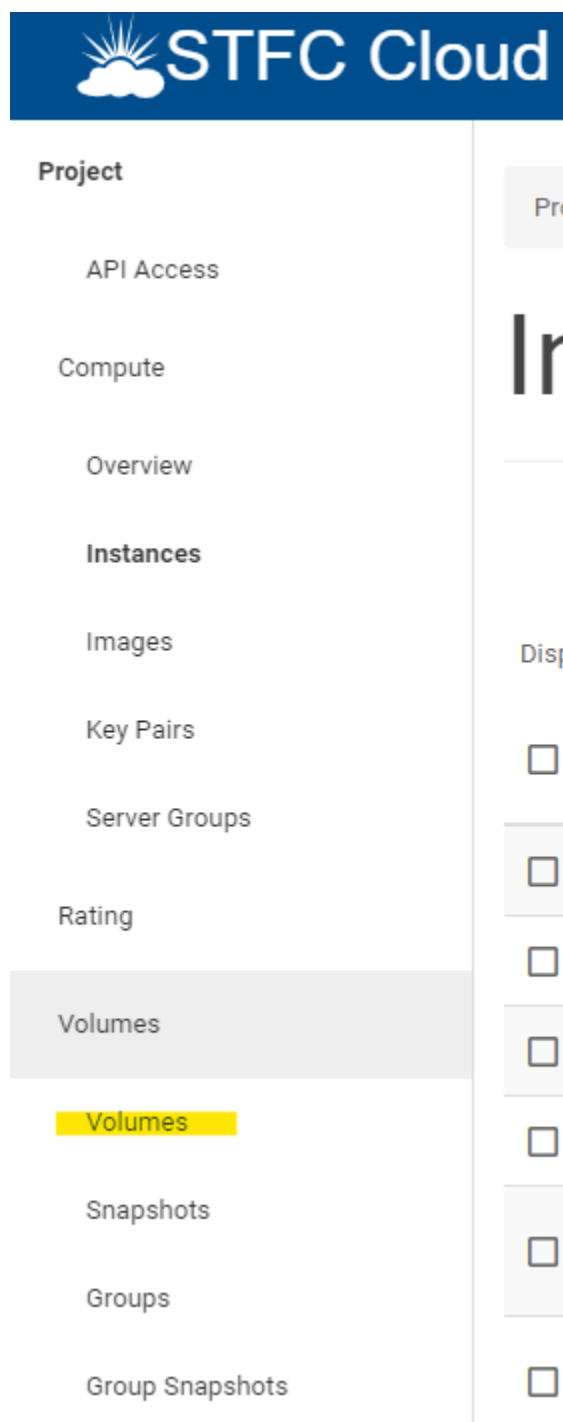
- \* *Web Interface*
- \* *Command-Line*
- *Deleting Volumes*
  - \* *Web Interface*
  - \* *Command-Line*

### 3.24.1 Create Volume

You can create a new volume using either the Web Interface or the CLI. You can also create volume from a snapshot. A snapshot is a mechanism where you to create a new image from a running services which serves as a templating and backup mechanism.

#### Web Interface

1. Log-in to the STFC cloud (<https://openstack.stfc.ac.uk/>)
2. In the Web Interface, Go to *Volumes* → *Volumes*



3. Click on *CREATE VOLUME*

# Volumes

---



+ CREATE VOLUME

↔ ACCEPT TRANSFER

🗑 DELETE VOLUMES

Displaying 3 items

4. Choose a name and size (in GiB),
5. (Optional) you can select to create a volume from a **snapshot** in the *volume source* drop-down

# Create Volume

Volume Name

testing-web

Description

Volume Source

NO SOURCE, EMPTY VOLUME

Type

\_\_DEFAULT\_\_

Size (GiB)

3

Availability Zone

ANY AVAILABILITY ZONE

Group

NO GROUP

Description:

Volumes are block devices that can be attached to instances.

Volume Type Description:

\_\_DEFAULT\_\_

Default Volume Type

Volume Limits

Total Gibibytes

of

GiB Used

Number of Volumes

of

Used

CANCEL

CREATE VOLUME

6. After creation, you will see the volume in the list

# Volumes

Filter

+ CREATE VOLUME

← ACCEPT TRANSFER

DELETE VOLUMES

Displaying 3 items

<input type="checkbox"/>	Name	Description	Size	Status	Group	Type	Attached To	Availability Zone	Bootable	Encrypted	Actions
<input type="checkbox"/>	testing-web	-	3GiB	Available	-	__DEFAULT__	ceph		No	No	EDIT VOLUME

## Command-Line

**Note:** See *Using OpenStack Command-line Interface* on how to set-up the command line client.

### Create simple volume

1. Run the following command in Terminal

```
openstack volume create --size <size-in-GiB> <name>
```

Example

```
$ openstack volume create --size 3 tutorial-volume
+-----+-----+
| Field          | Value                                     |
+-----+-----+
| attachments    | []                                       |
| availability_zone | ceph                                   |
| bootable       | false                                  |
| consistencygroup_id | None                                 |
| created_at     | 2021-12-10T16:59:52.000000           |
| description    | None                                  |
| encrypted      | False                                 |
| id             | 67c52d99-544a-4ea1-b8a7-ee added 7cb9df6 |
+-----+-----+
```

(continues on next page)

(continued from previous page)

```

↪ |
| multiattach          | False
↪ |
| name                 | tutorial-volume
↪ |
| properties           |
↪ |
| replication_status   | None
↪ |
| size                 | 3
↪ |
| snapshot_id          | None
↪ |
| source_volid         | None
↪ |
| status               | creating
↪ |
| type                 | __DEFAULT__
↪ |
| updated_at           | None
↪ |
| user_id              | 3ae4ecf4b9e0e66260b7aaebc2cc98aac3c95221e42f1cb49113ed751d8b9f2c
↪ |
+-----+-----+
↪ +

```

## Create image from snapshot/image

1. Find the ID of your snapshot

```

$ openstack volume snapshot list
+-----+-----+-----+-----+-----+-----+
↪ +-----+-----+-----+-----+-----+-----+
| ID                                     | Name                                     |
↪ +-----+-----+-----+-----+-----+-----+
| 76d51455-a5cd-478d-a93f-6e49b4108575 | testing-v-snapshot                     | None
↪ | available | 3 |
+-----+-----+-----+-----+-----+-----+
↪ +-----+-----+-----+-----+-----+-----+

```

2. Run the following command in Terminal

```
openstack volume create --snapshot <snapshot-id> --size <size> <name>
```

Example

```

$ openstack volume create --snapshot 76d51455-a5cd-478d-a93f-6e49b4108575 --size 3 test-
↪ cli-snapshot
+-----+-----+-----+-----+-----+-----+

```

(continues on next page)

(continued from previous page)

↩+		
	Field	Value
↩		
+	-----+	-----
↩+		
	attachments	[]
↩		
	availability_zone	ceph
↩		
	bootable	false
↩		
	consistencygroup_id	None
↩		
	created_at	2021-12-02T14:39:33.000000
↩		
	description	None
↩		
	encrypted	False
↩		
	id	2d61791d-5f52-46e1-81ac-05221c308fe8
↩		
	multiattach	False
↩		
	name	test-cli-snapshot
↩		
	properties	
↩		
	replication_status	None
↩		
	size	3
↩		
	snapshot_id	76d51455-a5cd-478d-a93f-6e49b4108575
↩		
	source_volid	None
↩		
	status	creating
↩		
	type	__DEFAULT__
↩		
	updated_at	None
↩		
	user_id	3ae4ecf4b9e0e66260b7aaebc2cc98aac3c95221e42f1cb49113ed751d8b9f2c
↩		
+	-----+	-----
↩+		

### 3.24.2 Attaching Volumes to an instance

You can only attach a volume to one instance at a time.

#### Web Interface

1. Log-in to the STFC cloud (<https://openstack.stfc.ac.uk/>)
2. In the Web Interface, Go to *Compute* → *Instances*
3. Click the drop-down menu on the right-hand side (in *Actions* column) and select *MANAGE ATTACHMENTS*

## Volumes

Displaying 4 items

<input type="checkbox"/>	Name	Description	Size	Status	Group	Type	Attached To	Availability Zone	Bootable	Encrypted	Actions
<input type="checkbox"/>	cli-new-volume	-	8GiB	Available	-	__DEFAULT__		ceph	No	No	EDIT VOLUME
<input type="checkbox"/>	testing-web	-	3GiB	Available	-	__DEFAULT__		ceph	No	No	EDIT VOLUME
<input type="checkbox"/>	testing	-	1GiB	Available	-	__DEFAULT__		ceph	No	No	EXTEND VOLUME MANAGE ATTACHMENTS

4. Select the right instance and press *ATTACH VOLUME*

# Manage Volume Attachments



Instance	Device	Actions
No items to display.		

## Attach To Instance

Attach to Instance

SELECT AN INSTANCE

CANCEL

ATTACH VOLUME

- Note the path in the *Attached To* column of the volume

<input type="checkbox"/>	Name	Description	Size	Status	Group	Type	Attached To	Availability Zone	Bootable	Encrypted	Actions
<input type="checkbox"/>	cli-ne										
<input type="checkbox"/>	w-volu	-	8GiB	Available	-	__DEFAULT__		ceph	No	No	EDIT VOLUME
<input type="checkbox"/>	testing	-	3GiB	In-use	-	__DEFAULT__	/dev/vdb n test-resiz e-rebuild	ceph	No	No	EDIT VOLUME

## Command-Line

**Note:** See [Using OpenStack Command-line Interface](#) on how to set-up the command line client.

- Get the Server ID (Instances) and Volume ID (Volume) using command

```
$ openstack server list
+-----+-----+-----+-----+-----+
| ID | Name | Status | Networks |
+-----+-----+-----+-----+
|  |  |  |  |
+-----+-----+-----+-----+
| ID | Image | Name | Status | Networks |
+-----+-----+-----+-----+
|  |  |  |  |  |
+-----+-----+-----+-----+
|  |  |  |  |  |
+-----+-----+-----+-----+
```

(continues on next page)

(continued from previous page)

```

↪ Flavor      |
+-----+-----+-----+-----+
↪ -----+
↪ -----+
| 6b2bedc4-9d8e-4bf3-be63-1dd49bc2e188 | test-resize-rebuild | ACTIVE | ↪
↪ Internal=172.16.102.207 | ubuntu-focal-20.04-gui | ↪
↪ | c3.small |
+-----+-----+-----+-----+
↪ -----+
↪ -----+
$ openstack volume list
+-----+-----+-----+-----+
↪ -----+
| ID | Name | Status | Size | Attached | ↪
↪ to |
+-----+-----+-----+-----+
↪ -----+
| 2d61791d-5f52-46e1-81ac-05221c308fe8 | test-cli-snapshot | available | 3 | ↪
↪ |
+-----+-----+-----+-----+
↪ -----+

```

## 2. Run

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

### Example

```

$ openstack server add volume 6b2bedc4-9d8e-4bf3-be63-1dd49bc2e188 2d61791d-5f52-46e1-
↪ 81ac-05221c308fe8 --device /dev/vdb
+-----+-----+
| Field | Value |
+-----+-----+
| ID | 2d61791d-5f52-46e1-81ac-05221c308fe8 |
| Server ID | 6b2bedc4-9d8e-4bf3-be63-1dd49bc2e188 |
| Volume ID | 2d61791d-5f52-46e1-81ac-05221c308fe8 |
| Device | /dev/vdb |
+-----+-----+

```

### 3.24.3 Accessing the volume

In order to access the volume you must also mount the volume in the VM.

1. Log-in to the attached instance using SSH
2. Use `lsblk` to confirm the device path (usually type disk). The value shown in OpenStack can be inaccurate.

```

$ lsblk
NAME        MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
loop0       7:0    0 73.1M  1 loop /snap/lxd/21902
loop1       7:1    0 55.4M  1 loop /snap/core18/2128

```

(continues on next page)

(continued from previous page)

```

loop3      7:3      0 72.6M 1 loop /snap/lxd/21750
loop4      7:4      0 61.9M 1 loop /snap/core20/1169
loop5      7:5      0 32.5M 1 loop /snap/snapd/13640
loop7      7:7      0 42.2M 1 loop /snap/snapd/14066
loop8      7:8      0 55.5M 1 loop /snap/core18/2253
loop9      7:9      0 61.9M 1 loop /snap/core20/1242
sr0        11:0      1 470K 0 rom /mnt/context
vda        252:0      0 20G 0 disk
├─vda1     252:1      0 19.9G 0 part /
├─vda14    252:14     0 4M 0 part
└─vda15    252:15     0 106M 0 part /boot/efi
vdc        252:32     0 3G 0 disk

```

3. (Optional, only for new volume) Format the volume (we use `ext4` here and assume the attach point is `/dev/vdc`) (Formatting will **wipe** your data):

```
sudo mkfs.ext4 /dev/vdc
```

4. Mount the volume (we use the folder `/mnt/test-volume` as example)

```
sudo mkdir /mnt/test-volume
```

5. Add this mount point to `/etc/fstab`, so it will be mounted automatically on startup

```
sudo vim /etc/fstab
```

6. Add/edit the following line:

```
/dev/vdc /mnt/test-volume ext4 defaults 0 0
```

7. You still need to manually mount it now

```
sudo mount /mnt/test-volume
```

8. (Optional) You may also want to change the permission of the directory using `chmod` to enable read/write without `sudo`

### 3.24.4 Detaching Volume

You can detach a volume using both command-line and web interface.

#### Web Interface

1. Log-in to the STFC cloud (<https://openstack.stfc.ac.uk/>)
2. In the Web Interface, Go to *Compute* → *Instances*
3. Click the drop-down menu on the right-hand side (in *Actions* column) and select *MANAGE ATTACHMENTS*

## Manage Volume Attachments



DETACH VOLUMES

Displaying 1 item

<input type="checkbox"/> Instance	Device	Actions
<input type="checkbox"/> test-resize-rebuild	/dev/vdb	DETACH VOLUME

Displaying 1 item

CANCEL

4. Click *DETACH VOLUME*

## Manage Volume Attachments



DETACH VOLUMES

Displaying 1 item

<input type="checkbox"/> Instance	Device	Actions
<input type="checkbox"/> test-resize-rebuild	/dev/vdb	DETACH VOLUME

Displaying 1 item

CANCEL

## Command-line

**Note:** See *Using OpenStack Command-line Interface* on how to set-up the command line client.

1. Run

```
openstack server remove volume <server-id> <volume-id>
```

Example

```
$ openstack server remove volume 6b2bedc4-9d8e-4bf3-be63-1dd49bc2e188 2d61791d-5f52-46e1-81ac-05221c308fe8
```

### 3.24.5 Create Volume Snapshot

#### Web Interface

1. Log-in to the STFC cloud (<https://openstack.stfc.ac.uk/>)
2. In the Web Interface, Go to *Volumes* → *Volumes*
3. Click the drop-down menu on the right-hand side (in *Actions* column) and select *CREATE SNAPSHOT*

# Volumes

<div> <input type="text" value="Filter"/> <input type="button" value="+ CREATE VOLUME"/> <input type="button" value="ACCEPT TRANSFER"/> <input type="button" value="DELETE VOLUMES"/> </div>											
Displaying 5 items											
<input type="checkbox"/>	Name	Description	Size	Status	Group	Type	Attached To	Availability Zone	Bootable	Encrypted	Actions
<input type="checkbox"/>	test-cli	-	3GiB	In-use	-	__DEFAULT__	/dev/vdb o n test-resiz e-rebuild	ceph	No	No	<div> <div>EDIT VOLUME</div> <div>MANAGE ATTACHMENTS</div> <div>CREATE SNAPSHOT</div> </div>
	hot										
	cli-new										

4. Give it a name and click *DETACH VOLUME*

×

# Create Volume Snapshot

Snapshot Name

Description

Description:

From here you can create a snapshot of a volume.

## Snapshot Limits

Total Gibibytes

of . GiB Used

Number of Snapshots

of . Used

CANCEL

CREATE VOLUME SNAPSHOT

## Command-Line

**Note:** See *Using OpenStack Command-line Interface* on how to set-up the command line client.

1. Find the ID of the volume with `openstack volume list`:

```
$ openstack volume list
+-----+-----+-----+-----+-----+
| ID                                     | Name           | Status   | Size | Attached to |
+-----+-----+-----+-----+-----+
| 03a5bb45-6c28-406d-8cd7-7fac5b63bdeb | cli-new-volume | available | 8    |              |
+-----+-----+-----+-----+-----+
```

2. Run

```
openstack volume snapshot create --volume <volume-id> <name>
```

Example

```
$ openstack volume snapshot create --volume 8e20dbdd-16ee-40e9-84ed-971c12104b98 testing-
↪v-snapshot
```

Field	Value
created_at	2021-12-02T14:34:48.718892
description	None
id	76d51455-a5cd-478d-a93f-6e49b4108575
name	testing-v-snapshot
properties	
size	3
status	creating
updated_at	None
volume_id	8e20dbdd-16ee-40e9-84ed-971c12104b98

### 3.24.6 Deleting Volumes

**Warning:** You should always refer to *Create Volume Snapshot* as this process is **not reversible** and may result in data loss.

#### Web Interface

1. Log-in to the STFC cloud (<https://openstack.stfc.ac.uk/>)
2. In the Web Interface, Go to *Volumes* → *Volumes*
3. Select the volume you wish to delete and click *DELETE VOLUME*
4. Confirm by clicking *DELETE VOLUMES*

#### Command-Line

**Note:** See *Using OpenStack Command-line Interface* on how to set-up the command line client.

1. Find the volume ID with `openstack volume list`

```
$ openstack volume list
```

ID	Name	Status	Size	Attached
d04f368d-7d60-4843-8f76-dbe61e73f9ee	delete-v-cli	available	1	

2. Run

```
openstack volume delete <volume-id>
```

Example

```
$ openstack volume delete d04f368d-7d60-4843-8f76-dbe61e73f9ee
```

## 3.25 Security Group

A **security group** acts as a virtual **firewall** for servers and other resources on a network with a set of **security group rules** that dictates the network access for those resources.

### Table of Contents

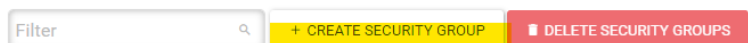
- *Security Group*
  - *Create Security Group*
    - \* *Web Interface*
    - \* *Command-Line*
  - *Delete Security Group*
    - \* *Web Interface*
    - \* *Command-Line*
  - *Security Group Rule Management*
    - \* *Add Security Group Rule*
      - *Web Interface*
      - *Command-Line*
    - \* *Delete Security Group Rule*
      - *Web Interface*
      - *Command-line*

### 3.25.1 Create Security Group

#### Web Interface

1. Log-in to the STFC cloud (<https://openstack.stfc.ac.uk/>)
2. In the Web Interface, Go to *Network* → *Security Groups*
3. Click *CREATE SECURITY GROUP*

## Security Groups



4. Give it a name and optionally a description and click *CREATE SECURITY GROUP*

## Create Security Group ✕

Name \*

Description

### Description:

Security groups are sets of IP filter rules that are applied to network interfaces of a VM. After the security group is created, you can add rules to the security group.

CREATE SECURITY GROUP

5. See Security Group Rule Management for how to edit the security group rules.

### Command-Line

**Note:** See *Using OpenStack Command-line Interface* on how to set-up the command line client.

1. Run

```
openstack security group create [--description <description>] <name>
```

Example

```
$ openstack security group create test-tutor
```

```
+-----+-----+
| Field          | Value                                     |
+-----+-----+
| created_at     | 2021-12-06T12:29:10Z                    |
+-----+-----+
| description    | test-tutor                              |
+-----+-----+
```

(continues on next page)

(continued from previous page)

```

↪
↪
| id | e9c8ffad-2b5c-4a47-9887-9dc0ebffa8b5
↪
↪
| location | Munch({'cloud': '', 'region_name': 'RegionOne', 'zone': None,
↪ 'project': Munch({'id': '80ab2bd11e5f46bf96bf47658d07499d', 'domain_id':
↪ '38372510d9bb4ac7916178b062d387de', 'domain_name': None})}) |
| name | test-tutor
↪
↪
| project_id | 80ab2bd11e5f46bf96bf47658d07499d
↪
↪
| revision_number | 1
↪
↪
| rules | created_at='2021-12-06T12:29:10Z', direction='egress', ethertype=
↪ 'IPv4', id='d23c7234-8f83-4307-84df-519aad87aa8c', updated_at='2021-12-06T12:29:10Z'
↪
| | created_at='2021-12-06T12:29:10Z', direction='egress', ethertype=
↪ 'IPv6', id='fc47d981-2368-403b-82bd-9ce6cdd9aa0d', updated_at='2021-12-06T12:29:10Z'
↪
| stateful | None
↪
↪
| tags | []
↪
↪
| updated_at | 2021-12-06T12:29:10Z
↪
↪
+-----+-----+
↪ -----
↪ -----

```

### 3.25.2 Delete Security Group

#### Web Interface

1. Log-in to the STFC cloud (<https://openstack.stfc.ac.uk/>)
2. In the Web Interface, Go to *Network* → *Security Groups*
3. Select the security groups that you wish to delete and click *DELETE SECURITY GROUPS*

# Security Groups

Displaying 18 items

<input type="checkbox"/>	Name	Security Group ID	Description	Actions
<input checked="" type="checkbox"/>	1-group-rule-management	c7e90937-af11-484c-8a21-90bea10e1407		MANAGE RULES

4. Confirm by clicking *DELETE SECURITY GROUPS*

## Confirm Delete Security Groups

**Warning:** This action cannot be undone.

You have selected:

- "1-group-to-delete"



### Command-Line

**Note:** See *Using OpenStack Command-line Interface* on how to set-up the command line client.

1. Find the security group ID with `openstack server list`

```
$ openstack security group list
```

ID	Project	Name	Description
d20fd393-8193-4068-a181-cf4861f112f7	80ab2bd11e5f46bf96bf47658d07499d	1-group-to-delete	

2. Run

```
openstack security group delete <security-group-id>
```

Example

```
$ openstack security group delete d20fd393-8193-4068-a181-cf4861f112f7
```

### 3.25.3 Security Group Rule Management

Security Group Rules define which traffic is allowed to instances assigned to the security group. A security group rule consists of three main parts:

- **Rule:** You can specify the desired rule template or use custom rules, the options are Custom TCP Rule, Custom UDP Rule, or Custom ICMP Rule.
- **Open Port/Port Range:** For TCP and UDP rules you may choose to open either a single port or a range of ports. Selecting the “Port Range” option will provide you with space to provide both the starting and ending ports for the range. For ICMP rules you instead specify an ICMP type and code in the spaces provided.
- **Remote:** You must specify the source of the traffic to be allowed via this rule. You may do so either in the form of an:
  - IP address block (CIDR)
  - Source group (Security Group) Selecting a security group as the source will allow any other instance in that security group access to any other instance via this rule.

#### Add Security Group Rule

##### Web Interface

1. Log-in to the STFC cloud (<https://openstack.stfc.ac.uk/>)
2. In the Web Interface, Go to *Network* → *Security Groups*
3. Click *MANAGE RULES*

## Security Groups

<div> <input type="text" value="Filter"/> <input type="button" value="+ CREATE SECURITY GROUP"/> <input type="button" value="DELETE SECURITY GROUPS"/> </div>			
Displaying 18 items			
<input type="checkbox"/>	Name	Security Group ID	Description
<input type="checkbox"/>	1-group-rule-management	c7e90937-af11-484c-8a21-90bea10e1407	
			<input type="button" value="MANAGE RULES"/>

4. Click *ADD RULE*

Project / Network / Security Groups / Manage Security Group Rule...

# Manage Security Group Rules: 1-group-rule-management (c7e90937-af11-484c-8a21-90bea10e1407)

---

+ ADD RULE

■ DELETE RULES

Displaying 2 items

5. Input the following:

- Rule: TCP, UDP or ICMP
- Direction
  - Ingress
  - Egress
- Open Port:
  - Port
  - Port Range
- Port
  - the port number
  - You will see From port and To port filed if you selected PORT RANGE
- Remote (Source IP)
  - CIDR
    - \* Input the IP range
  - SECURITY GROUP
    - \* Select the Security group

# Add Rule



Rule \*

CUSTOM TCP RULE

Description ?

Direction

INGRESS

Open Port \*

PORT

Port ?

Remote ?

CIDR

CIDR ?

0.0.0.0/0

## Description:

Rules define which traffic is allowed to instances assigned to the security group. A security group rule consists of three main parts:

**Rule:** You can specify the desired rule template or use custom rules, the options are Custom TCP Rule, Custom UDP Rule, or Custom ICMP Rule.

**Open Port/Port Range:** For TCP and UDP rules you may choose to open either a single port or a range of ports. Selecting the "Port Range" option will provide you with space to provide both the starting and ending ports for the range. For ICMP rules you instead specify an ICMP type and code in the spaces provided.

**Remote:** You must specify the source of the traffic to be allowed via this rule. You may do so either in the form of an IP address block (CIDR) or via a source group (Security Group). Selecting a security group as the source will allow any other instance in that security group access to any other instance via this rule.

CANCEL

ADD

6: Click *ADD*

## Command-Line

### 1. Check existing security group using `openstack security group list`

```
$ openstack security group list
```

ID	Project	Name	Description
c7e90937-af11-484c-8a21-90bea10e1407	80ab2bd11e5f46bf96bf47658d07499d	1-group-rule-management	

```
$ openstack security group rule list 1-group-rule-management
```

ID	IP Protocol	Ethertype	IP Range	Port Range	Direction	Remote Security Group	Remote Address Group
33cd09ee-0e9b-4ce3-b028-7e24f9604431	None	IPv6	::/0		egress	None	
7969c6ae-dec0-4071-935e-6a81e7d8ec5c	None	IPv4	0.0.0.0/0		egress	None	

### 3. Create security group rule

Table 3: Title

Argument	Description	Example
<security-group-name>	Name of the security group	1-group-rule-management
<rule>	The protocol: TCP, UDP or ICMP	tcp
<port-range>	The range of ports to apply the rule: <code>from_port:to_port</code>	89:90
<ip-range>	The source IP range for the rule	0.0.0.0/0
<source-security-group>	Name of the source security group to allow access	9200-Elastic-Search
[--ingress   --egress]	Ingress rule or egress rule (default is --ingress)	--egress

#### 3.a Security group rule based on traffic source IP

```
openstack security group rule create <security-group-name> --protocol <rule> --dst-port
<port-range> --remote-ip <ip-range> [--ingress | --egress]
```

## Example

```
$ openstack security group rule create 1-group-rule-management --protocol tcp --dst-port 89:90 --remote-ip 0.0.0.0/0
$ openstack security group rule list 1-group-rule-management
```

ID	Direction	Remote Security Group	Remote Address Group	IP Protocol	Ethertype	IP Range	Port
33cd09ee-0e9b-4ce3-b028-7e24f9604431	egress	None	None	None	IPv6	::/0	
7969c6ae-dec0-4071-935e-6a81e7d8ec5c	egress	None	None	None	IPv4	0.0.0.0/0	
7f4e2c68-a369-4be8-8491-561cccffc90c	ingress	None	None	tcp	IPv4	0.0.0.0/0	89:90

## 3.b Security group rule based on source security group

```
openstack security group rule create <security-group-name> --protocol <rule> --dst-port <port-range> --remote-group <source-security-group> [--ingress | --egress]
```

## Example

```
$ openstack security group rule create 1-group-rule-management --protocol tcp --dst-port 1021:1057 --remote-group 9200-Elastic-Search --egress
$ openstack security group rule list 1-group-rule-management
```

ID	Direction	Remote Security Group	Remote Address Group	IP Protocol	Ethertype	IP Range	Port
33cd09ee-0e9b-4ce3-b028-7e24f9604431	egress	None	None	None	IPv6	::/0	
7969c6ae-dec0-4071-935e-6a81e7d8ec5c	egress	None	None	None	IPv4	0.0.0.0/0	
f6bc9897-cc2c-40d8-ae1-3efe5107c394	egress	6f559f5d-9e92-4603-8226-de37cb39dcd8	None	tcp	IPv4	0.0.0.0/0	1021:1057

## Delete Security Group Rule

### Web Interface

1. Log-in to the STFC cloud (<https://openstack.stfc.ac.uk/>)
2. In the Web Interface, Go to *Network* → *Security Groups*
3. Click *MANAGE RULES*

# Security Groups

Displaying 18 items

<input type="checkbox"/>	Name	Security Group ID	Description	Actions
<input type="checkbox"/>	1-group-rule-management	c7e90937-af11-484c-8a21-90bea10e1407		<input type="button" value="MANAGE RULES"/>

4. Select the rules you want to delete and click *DELETE RULE*

## Manage Security Group Rules: 1-group-rule-management (c7e90937-af11-484c-8a21-90bea10e1407)

Displaying 4 items

<input type="checkbox"/>	Direction	Ether Type	IP Protocol	Port Range	Remote IP Prefix	Remote Security Group	Description	Actions
<input type="checkbox"/>	Egress	IPv4	Any	Any	0.0.0.0/0	-	-	<input type="button" value="DELETE RULE"/>
<input checked="" type="checkbox"/>	Egress	IPv4	TCP	1021 - 1057	-	9200-Elastic-Search	-	<input type="button" value="DELETE RULE"/>
<input type="checkbox"/>	Egress	IPv6	Any	Any	:::/0	-	-	<input type="button" value="DELETE RULE"/>
<input type="checkbox"/>	Ingress	IPv4	TCP	89 - 90	0.0.0.0/0	-	-	<input type="button" value="DELETE RULE"/>

5. Click *DELETE RULE* to confirm delete

## Confirm Delete Rule



**Warning:** This action cannot be undone.

You have selected:

- "ALLOW IPv4 1021-1057/tcp from 9200-Elastic-Search"

## Command-line

1. Get the security group name using `openstack security group list`

```
$ openstack security group list
```

ID	Project	Name	Description
c7e90937-af11-484c-8a21-90bea10e1407	80ab2bd11e5f46bf96bf47658d07499d	1-group-rule-management	

2. Get the ID of the rule with `openstack security group rule list <security-group-name>`

```
$ openstack security group rule list 1-group-rule-management
```

ID	IP Protocol	Ethertype	IP Range	Port Range	Direction	Remote Security Group	Remote Address Group
33cd09ee-0e9b-4ce3-b028-7e24f9604431	None	IPv6	::/0		egress	None	None
7969c6ae-dec0-4071-935e-6a81e7d8ec5c	None	IPv4	0.0.0.0/0		egress	None	None
7f4e2c68-a369-4be8-8491-561cccffc90c	tcp	IPv4	0.0.0.0/0	89:90	ingress	None	None

3. Run

```
openstack security group rule delete <security-group-rule-id>
```

```
$ openstack security group rule delete 7f4e2c68-a369-4be8-8491-561cccffc90c
$ openstack security group rule list 1-group-rule-management
```

ID	IP Protocol	Ethertype	IP Range	Port Range	Direction	Remote Security Group	Remote Address Group
33cd09ee-0e9b-4ce3-b028-7e24f9604431	None	IPv6	::/0		egress	None	None
7969c6ae-dec0-4071-935e-6a81e7d8ec5c	None	IPv4	0.0.0.0/0		egress	None	None

## 3.26 Ansible and terraform on STFC cloud

### Table of Contents

- *Ansible and terraform on STFC cloud*
  - *Why Infrastructure as code (IaC)?*
  - *Tools for IaC*
    - \* *Terraform V.S. Ansible*
    - \* *Terraform*
      - *Installing Terraform*
      - *Setting up provider*
      - *Applying config via terraform*
      - *Reference*
    - \* *Ansible*
      - *Installing ansible*
      - *Setting up target*
      - *Preparing key file*
      - *Setting up playbook*
      - *Running ansible-playbook*
      - *Reference*

### 3.26.1 Why Infrastructure as code (IaC)?

IaC allows users to avoid manual configuration of environments and enforce consistency by representing the desired state of their environments as code. They are repeatable and prevent runtime issues caused by configuration drift or missing dependencies.

Always treat your resources (VMs etc.) like cattle, not pets, they are all replaceable, and you shouldn't directly SSH into the machine and make specific changes.

### 3.26.2 Tools for IaC

We recommend two tools for this purpose. *Terraform* and *Ansible*.

#### Terraform V.S. Ansible

Always choose the tool you need for the job, Terraform is usually used to create new infrastructure, and Ansible is used to configure and manage the infrastructure.

#### Terraform

Terraform is an infrastructure as code (IaC) tool for creating, maintaining and decommissioning large data center infrastructure. The configurations are specified in a declarative language, where it will then determine the steps to transform infrastructures to the new desired state.

Advantages: \* Plan: A useful feature of Terraform is the `terraform plan` where it will perform a dry run and show the changes required without actually performing them. \* It is independent of cloud provider. \* It can integrate multiple cloud providers. This allows a hybrid cloud approach that provision resources across multiple providers.

#### Installing Terraform

On a Ubuntu machine run the following command

```
# register key
curl -fsSL https://apt.releases.hashicorp.com/gpg | sudo apt-key add -
# add repo
sudo apt-add-repository "deb [arch=$(dpkg --print-architecture)] https://apt.releases.
↳hashicorp.com $(lsb_release -cs) main"
# install with apt
sudo apt install terraform
```

#### Setting up provider

1. Create a directory and cd to the directory
2. Create a file named `providers.tf` with the following content. This will enable the *OpenStack* provider.

Listing 1: providers.tf

```
# Define required providers
terraform {
  required_version = ">= 0.14.0"
  required_providers {
    openstack = {
      source = "terraform-provider-openstack/openstack"
      version = "~> 1.35.0"
    }
  }
}

provider "openstack" {}
```

3. Declare the cloud information by source `<openstack-rc-file>`.

---

**Note:** See *Using OpenStack Command-line Interface* on login as using `provider.tf` is not secure

---

4. Run `terraform init`

```
$ terraform init

Initializing the backend...

Initializing provider plugins...
- Reusing previous version of terraform-provider-openstack/openstack from the dependency_
↳lock file
- Using previously-installed terraform-provider-openstack/openstack v1.35.0

Terraform has been successfully initialized!

You may now begin working with Terraform. Try running "terraform plan" to see
any changes that are required for your infrastructure. All Terraform commands
should now work.

If you ever set or change modules or backend configuration for Terraform,
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.
```

## Applying config via terraform

To create a configuration you need the `deploy.tf` for your configuration and `variables.tf` for the variables in the provider folder

I have create an example that will create a set of instances with a volume attached to it.

Listing 2: `variables.tf`

```
# create a SSH key-pair using other method first since terraform is not secure
variable "key_pair_name" {
    description = "key pair name"
    default    = "tutorial"
}

variable "http_sec_group" {
    description = "custom security group name"
    default    = "HTTP-ingress"
}

variable "network_name" {
    description = "The network to be used."
    default    = "Internal"
}

variable "instance_name" {
    description = "The Instance Name to be used."
```

(continues on next page)

(continued from previous page)

```

    default = "tutorial-machine"
}

variable "image_id" {
    #find with: openstack image list
    description = "The image ID to be used."
    default = "622cb70d-c88c-4dc4-99e6-df8b8f9965d7"
}

variable "flavor_id" {
    #find with: openstack flavor list
    description = "The flavor id to be used."
    default = "022ace2c-5247-4bdc-8929-81d129cc69bf"
}

# have to put http sec group here as well
variable "security_groups" {
    description = "List of security group"
    type = list
    default = ["default", "HTTP-ingress"]
}

variable "instance_num" {
    description = "The keypair public key."
    default = 1
}

variable "volume_name" {
    description = "name of volume"
    default = "tutorial_volume"
}

variable "volume_size" {
    description = "size of volume"
    default = 3
}

```

Listing 3: delpoy.tf

```

# create a security group named secgroup_1
resource "openstack_networking_secgroup_v2" "secgroup_1" {
    # you can access the variables using var.<variable-name>
    name = var.http_sec_group
    description = "My neutron security group"
}

# create a security group rules
resource "openstack_networking_secgroup_rule_v2" "secgroup_rule_1" {
    direction = "ingress"
    ethertype = "IPv4"
    protocol = "tcp"
    port_range_min = 80
}

```

(continues on next page)

(continued from previous page)

```

port_range_max    = 80
remote_ip_prefix  = "0.0.0.0/0"
# you can access the output attribute of other resources using ${<provider>.<name>.<attribute>}
security_group_id = "${openstack_networking_secgroup_v2.secgroup_1.id}"
}

#creating instances
resource "openstack_compute_instance_v2" "Instance" {
# the count will create a number of duplicates equal to the number
count = var.instance_num
name = format("%s-%02d", var.instance_name, count.index+1)
image_id = var.image_id
flavor_id = var.flavor_id
key_pair = var.key_pair_name
security_groups = var.security_groups
network {
    name = var.network_name
}
}

resource "openstack_blockstorage_volume_v2" "Volume" {
count = var.instance_num
name    = format("%s-%02d", var.volume_name, count.index+1)
size    = var.volume_size
}

resource "openstack_compute_volume_attach_v2" "attachments" {
count = var.instance_num
# if there are multiples of same resources the detail are stored in an array
instance_id = "${openstack_compute_instance_v2.Instance[count.index].id}"
volume_id   = "${openstack_blockstorage_volume_v2.Volume[count.index].id}"
}

# this will create a output which you can use
output "instance_ips" {
    value = {
        for instance in openstack_compute_instance_v2.Instance:
            instance.name => instance.access_ip_v4
    }
}

```

You should create ssh keys via the web interface or OpenStack clients and reference it in terraform since terraform will store configuration in plain text which will expose your ssh key.

After you have edited the files, you can run `terraform plan` or `terraform apply` to review changes and apply them to the cloud.

To change the variable you can either change the default in `variable.tf` or by passing the `-var "<variable-name>=<value>"` flag during `terraform plan` or `terraform apply`.

Example:

```
$ terraform plan -var "instance_num=2"
```

```
$ terraform apply -var "instance_num=2"
...
Apply complete! Resources: 8 added, 0 changed, 0 destroyed.

Outputs:

instance_ips = {
"tutorial-machine-01" = "172.16.100.135"
"tutorial-machine-02" = "172.16.101.124"
}
```

## Reference

You should refer to [Docs overview | terraform-provider-openstack/openstack](#) | [Terraform Registry](#) for details on how to use the providers

Table 4: Common providers

Provider	Function
openstack_compute_instance_v2	Create instance
openstack_networking_secgroup_v2	Create security group
openstack_networking_secgroup_rule_v2	Create security group rule
openstack_networking_network_v2	Create network
openstack_networking_floatingip_v2	Get a floating IP from allocated pool
openstack_compute_floatingip_associate_v2	Associate floating IP to instances
openstack_containerinfra_clustertemplate_v1	Create magnum cluster template
openstack_containerinfra_cluster_v1	Create magnum cluster
openstack_blockstorage_volume_v2	Create a new volume
openstack_compute_volume_attach_v2	Attach volume to instances
openstack_lb_loadbalancer_v2	Create Load balancer
openstack_lb_listener_v2	Create Load balancer listener
openstack_lb_pool_v2	Set method for load balance charge between instance (e.g. round robin)
openstack_lb_member_v2	Add instance to load balancer
openstack_lb_monitor_v2	Create health monitor for load balancer

## Ansible

Ansible is a Python-based IT system configuration automation tool that gained widespread use as a network automation system

Advantages: \* It uses YAML based playbook \* It is agentless, which means that you don't have to load anything to the target machine. \* Large community with open source and vendor support

## Installing ansible

```
sudo apt update
sudo apt install software-properties-common
sudo apt-add-repository ppa:ansible/ansible
sudo apt update
sudo apt install ansible
```

## Setting up target

Create a Host file refer to this [How to build your inventory — Ansible Documentation](#)

Listing 4: hosts

```
all:
# group of remote
hosts:
    172.16.100.135:
    172.16.101.124:
```

## Preparing key file

You should use the \*.pem file you downloaded when you created the key

Your ssh key must not be too open set permission using `sudo chmod 600 <path-to-key>`

## Setting up playbook

Here is an example of a simple playbook. you can refer to ansible documentations for different providers.

Listing 5: hosts

```
- hosts: all
tasks:
  - name: Print message
    debug:
      msg: Hello Ansible World
```

## Running ansible-playbook

```
ansible-playbook -i ./hosts --private-key <path-to-key> playbook.yml
```

Example

```
$ ansible-playbook -i ./hosts --private-key ./key.pem playbook.yml
PLAY [all]
  TASK [Gathering Facts]
```

(continues on next page)

(continued from previous page)

```

↳ *****
ok: [172.16.100.135]
ok: [172.16.101.124]

TASK [Print message]
↳ *****
ok: [172.16.100.135] => {
    "msg": "Hello Ansible World"
}
ok: [172.16.101.124] => {
    "msg": "Hello Ansible World"
}

PLAY RECAP
↳ *****
172.16.100.135      : ok=2    changed=0    unreachable=0    failed=0    skipped=0
↳   rescued=0     ignored=0
172.16.101.124      : ok=2    changed=0    unreachable=0    failed=0    skipped=0
↳   rescued=0     ignored=0

```

## Reference

[Ansible Documentation](#) — [Ansible Documentation](#)



**POLICIES**

## **4.1 Backup policy of VM's in openstack**

### **4.1.1 Overview**

Description of backup policy for cloud VMs and data on them

## **4.2 Patching policy of VM's in openstack**

### **4.2.1 Overview**

Description of the patching policy of virtual machines in the cloud

## **4.3 STFC Cloud Privacy Policy**

Updated: 17/01/2022

For the purposes of the Data Protection Act 1998 and the General Data Protection Regulation, the STFC Cloud act as the data controller in respect of any data or personal information which we hold about users of our service. The terms “STFC Cloud”, “we” or “us” refer to the STFC Cloud in this policy. Questions or requests for action relating to this policy or any electronic interactions with the STFC Cloud, should be emailed to: [cloud-support@stfc.ac.uk](mailto:cloud-support@stfc.ac.uk)

This policy outlines how we manage the STFC Cloud's information collection and distribution practices and gives notice to how we will use the information provided to us. This policy applies to anyone providing us with personal information in order to sign up to our mailing list or newsletter or to use the STFC Cloud service. Any necessary updates to this policy will be made as needed and with immediate effect.

The STFC Cloud only collects personal information where it is volunteered. We do not make use of Cookies or analytical tools using personal data. Any identifying information you provided to us, such as names, email addresses or other information will only be used only for the following purposes:

1. To provide service updates.
2. To seek continual improvement feedback on our services.
3. To broadcast (by electronic means) information we believe may be of interest.
4. To respond to requests for information, amendments or withdrawals.
5. To deliver our electronic newsletter.
6. To verify the information we hold is correct.

This information will be kept secure within STFC Cloud and will not be shared with third parties, except with explicit permission.

All those who have opted-in to receive information from us, including our email newsletter, will receive communications from us in electronic format. Requests to remove consent, receive a copy of the personal information we have collected, or to add, correct or request that we delete personal information should be submitted to: [cloud-support@stfc.ac.uk](mailto:cloud-support@stfc.ac.uk)

## 4.4 Service Level Agreement

The SCD Cloud is a service developed for use within Scientific Computing and the wider STFC for developing and supporting projects. SCD Cloud usage must comply with the Terms of Service. If you have any questions regarding the following, please contact [cloud-support@stfc.ac.uk](mailto:cloud-support@stfc.ac.uk)

By continuing to log in you accept the following Service Level for Virtual Machines (VMs):

### 4.4.1 Support

- Technical Support for the SCD Cloud is offered during working hours.
- Tickets will be responded to within 2 working days.
- Support for the end-users of services hosted on the SCD Cloud is not provided, this includes but is not limited to using the Operating System.

### 4.4.2 Availability

- We endeavour to minimize downtime of VMs however some is inevitable.
- Services built on the SCD Cloud where availability is important must be architected in such a way that they can tolerate VMs being unavailable.
- VMs are configured to update automatically although monthly reboots will be required.

### 4.4.3 Storage

- The storage underlying the SCD Cloud Volume (Cinder), Image (Glance), Object (Swift) and Fileshare (Manila) services is resilient for the purposes of availability.
- VMs on the SCD Cloud should not be used as a persistent Datastore and have no resiliency of storage.
- Boot from Volume VMs should be used where resiliency is required
- If you require more security for your data then arrange to back up your VMs.

#### 4.4.4 Maintenance

- We will endeavour to provide notification of scheduled downtimes of 2 working days.
- Regular maintenance will take place on Wednesday mornings between 10am and 12pm. The service should be considered at risk during this period.
- If a security patch requires it, you will normally be given 5 working days within which to reboot VMs, after which time a reboot will be forced.
- We reserve the right to reboot VMs without notice in order to preserve the integrity of the service.

### 4.5 Who does the system administration for VMs in a project

#### 4.5.1 Overview

Describes who is responsible for the system administration of VMs in a project.

### 4.6 Terms Of Service

This is a service under continual development. If you are unsure about ANY of the following conditions of use or how they apply to you or your intended use of this service then, BEFORE continuing, please contact the Cloud Service Managers at [cloud-support@stfc.ac.uk](mailto:cloud-support@stfc.ac.uk) who will be pleased to help.

By continuing to log in you agree to abide by the following conditions of use of this service:

1. You MUST comply with Organisational Information Security Policy particularly regarding the Roles and Responsibilities of System Administrators together with familiarising yourself with the supporting policy framework available at <https://stfc.ukri.org/about-us/how-we-are-governed/policies-standards/ict-acceptable-use-policy/> and <https://community.jisc.ac.uk/library/acceptable-use-policy> (or for STFC Staff at <https://staff.stfc.ac.uk/core/security/information/Pages/default.aspx>)
2. You MUST NOT, except by WRITTEN authorization from the Cloud Service Managers, disable or otherwise modify or degrade the configuration of installed system monitoring and management tools including but not limited to that of rsyslog, pakiti, ssh, yum and auto-updating
3. You UNDERSTAND that, whilst best effort is made to provide a stable platform, VMs may be subject to interruption and/or data loss at any time without notice (this is a development platform and should not be used for production services)
4. With respect to any SOFTWARE that you install you MUST ensure that all applicable license and terms and conditions of use are met
5. You MUST NOT use the service in any way related to providing any commercial service or running a private business
6. You MUST report any suspected or actual security incident or other misuse of the VM immediately to [cloud-support@stfc.ac.uk](mailto:cloud-support@stfc.ac.uk).
7. Credentials applicable to the VM (such as X509 host certificate and Kerberos private keys) MUST be obtained through the Cloud Service Managers. You MUST apply and maintain appropriate protection to prevent exposure or misuse for all such credentials and NOT export private keys or take any other action which would prejudice credential re-use in future VM instances.

8. You will be informed by email of any changes to these conditions and **MUST** inform the Cloud Service Managers immediately if you can no longer abide by the updated conditions. The latest conditions of these conditions are available at [here](#)
9. You **AGREE** that you can be held liable for any consequences of your failure to abide by these conditions of use including, but not limited to, the possible immediate termination of your VM without notice, reporting to your organisational management and, if thought to be appropriate, necessary law enforcement agencies.

## CONTACT DETAILS

### 5.1 Contact Support

For support, queries and requests please email us at [cloud-support@stfc.ac.uk](mailto:cloud-support@stfc.ac.uk)

### 5.2 Mailing list

You can subscribe to our mailing list at <https://www.jiscmail.ac.uk/cgi-bin/webadmin?A0=STFC-CLOUD> to receive notifications from the Cloud team about any upcoming interventions, issues and changes in documentation.

### 5.3 Slack

You can join the STFC Cloud Slack to <https://stfc-cloud.slack.com/> to chat with the Cloud team, share feedback and message other users.

This is also the fastest way to get any notifications about the service. If you are unable to sign up, please email [cloud-support@stfc.ac.uk](mailto:cloud-support@stfc.ac.uk) and we will add your email domain to the white list.

---

**Important:** Please use the email provided (*Contact Support*) for requests, support and queries instead of messaging cloud team members directly on Slack.

Email requests are actively monitored and prioritised over direct Slack messages, which may be ignored or receive delayed responses. Emails also allows us to assign issues to the most appropriate team member to resolve your issue quickly.

---



**Contents**

- *FAQS*
  - *How do I get a new project in the STFC Cloud*
  - *How do I get support for the STFC cloud?*
  - *How do I get access to the cloud?*
  - *Which Web browsers are supported for the web user interfaces ?*
  - *How do I connect to my VM using ssh?*
  - *How do I copy files to and from my VM and between VM's?*
  - *How do I run a command on lots of my VM's at the same time?*
  - *When I use "sudo" command on a VM I created, it says <my username> not in Sudoers - Help!*
  - *How do I get a remote X session displayed from a VM?*
  - *How do I get a remote linux desktop GUI display on my VM on my windows desktop?*
  - *How can I view an Application on my Linux VM that requires some graphics acceleration and uses OpenGL?*
  - *How do I create a VM with GPU capabilities in my project?*
  - *How do I create a VM with shared disk space to my other VMs in my project?*
  - *How do I connect to another VM in my project which I did not create?*
  - *What do I do if a new VM that I create does not work?*
  - *How do I access my Virtual machines across the internet?*
  - *Is there a default firewall policy on my project (and what does it say ?)*
  - *How do I avoid Docker Hub Rate limits?*
  - *I have a Database listening on its default port, but I can't connect to it remotely - Why?*
  - *A VM machine I was using has died - How can I get the data off of it?*
  - *What are the current machine "flavors"? Can I have a machine that looks like a flavor you don't have?*
  - *Can you provide operating system "X" on Openstack?*
  - *What sort of CPU performance should I expect?*

- *What sort of disk I/O performance should I expect locally on a VM?*
- *What sort of network bandwidth should I expect on a VM?*
- *Can you recover a VM I accidentally deleted?*
- *How do I login to the “admin” interface?*
- *How do I obtain a host certificate for my Openstack virtual machine?*
- *My host seems to have rebooted since last time I logged in - why?*
- *What are the default DNS servers for VMs on Openstack?*

## 6.1 How do I get a new project in the STFC Cloud

As a bare minimum, the following should be specified:-

1. Project name (what you want your project to be called)
2. Project description (a brief description of the project - not essential, but useful if other similar named projects)
3. Users - their full names, and e-mail addresses. If they have internal STFC Federal ID logins, these are useful too.
4. Types of machines you use and how many (This is so that we can assign a resource quota for the project, how many CPUs, memory, disk space .etc.)
5. If the project either now, or in the known future, needs to host internet accessible services - what those services are, their purpose and how many IP addresses that are externally facing will be required.

## 6.2 How do I get support for the STFC cloud?

E-mail the Cloud support team on [cloud-support@stfc.ac.uk](mailto:cloud-support@stfc.ac.uk) .

There is no telephone support number at this time.

## 6.3 How do I get access to the cloud?

Contact the Cloud support team on [cloud-support@stfc.ac.uk](mailto:cloud-support@stfc.ac.uk) . For a new project see *How do I get a new project in the STFC Cloud*.

If this is about an existing project please email with the project and account name you would like to add access for.

Once you have these, you can connect using <https://cloud.stfc.ac.uk> for a simple user interface, or <http://openstack.stfc.ac.uk> for advanced users.

## 6.4 Which Web browsers are supported for the web user interfaces ?

The latest Firefox or Google Chrome browsers are recommended.

## 6.5 How do I connect to my VM using ssh?

By default VMs will be ssh-accessible from within the STFC network.

If you are external to STFC we will usually open ssh to one of your floating IP addresses whilst setting up your project.

If you are prompted for the password, ensure that your [private key is available](#) and that you have used the right username. For example `ubuntu@<your_ip>` on Ubuntu instances.

## 6.6 How do I copy files to and from my VM and between VM's?

The simplest way is with SCP, this is a default tool within Linux distributions and various versions are available for Windows.

Here are two examples of copying a file from your home directory to /tmp would be:

```
scp ~/my_file.txt <username>@<ip_addr>:/tmp
```

```
scp ~/my_file.txt ubuntu@example.com:/tmp/my_renamed_file.txt
```

## 6.7 How do I run a command on lots of my VM's at the same time?

ClusterSSH is a good way of doing this and is pretty straightforward if you are using a Linux desktop. Under Windows it is more complicated.

## 6.8 When I use “sudo” command on a VM I created, it says <my username> not in Sudoers - Help!

Occasionally this happens to a VM if something goes wrong or is too slow during the boot process.

In most cases a reboot of the VM from within OpenStack will fix this. If it doesn't please email [cloud-support@stfc.ac.uk](mailto:cloud-support@stfc.ac.uk)

## 6.9 How do I get a remote X session displayed from a VM?

This is not recommended for systems where untrusted parties have SSH access.

Enabling X forwarding comes with additional security considerations, as authentication spoofing and verification attacks can occur. [See the manual for additional details](#)

X11 Forwarding must be enabled on the remote server first. In `/etc/ssh/sshd_config` set `X11Forwarding Yes`, if it is not already enabled and restart the sshd service.

To enable X11 forwarding within the SSH session, simply add `-X` to your SSH command. `-C` is highly recommended for traffic routed via external networks. Support for forwarding is application specific, but most should work.

The following would enable compression and X11 forwarding:

```
ssh -CX foo@example.com
```

## 6.10 How do I get a remote linux desktop GUI display on my VM on my windows desktop?

If you have a GUI installed and running on your VM then you can use the console view:

- Navigate to <https://openstack.stfc.ac.uk> (or <https://cloud.stfc.ac.uk> within STFC)
- Click the dropdown next to the instance you would like to view
- Select console to view the video output

Otherwise you can install a `vncserver` and VNC client to view.

## 6.11 How can I view an Application on my Linux VM that requires some graphics acceleration and uses OpenGL?

Please contact us for further support.

## 6.12 How do I create a VM with GPU capabilities in my project?

If you have `g*` flavors available in your project then you can create a GPU accelerated VM. See *flavors for details on the GPU types available*.

If you don't have `g*` flavors then you can contact [cloud-support@stfc.ac.uk](mailto:cloud-support@stfc.ac.uk) to request access.

## 6.13 How do I create a VM with shared disk space to my other VMs in my project?

Documentation coming soon.

## 6.14 How do I connect to another VM in my project which I did not create?

In most cases we expect users to configure access to their VMs as required. If this hasn't been possible then please contact [cloud-support@stfc.ac.uk](mailto:cloud-support@stfc.ac.uk) and we can help with this.

For adding members to a project, so that they can modify cloud resources see *How do I get access to the cloud?*

## 6.15 What do I do if a new VM that I create does not work?

Please contact us as [cloud-support@stfc.ac.uk](mailto:cloud-support@stfc.ac.uk)

## 6.16 How do I access my Virtual machines across the internet?

You will need to contact [cloud-support@stfc.ac.uk](mailto:cloud-support@stfc.ac.uk) to request a firewall hole for your floating IP. We will then conduct some security checks and help you through this process.

Once the exception is added to your floating ip(s) you will need to add create and associate a security group with the exception to your instance:

To create a new security group with one or more ports:

- Open Network, Security Groups
- Create a new Security Group and enter a name (such as *HTTP + HTTPS*)
- Click next, leave the egress rules (as this allows traffic out) and add a rule per port
- Ensure Ingress is selected, and enter the port number.

To associate a new or existing security group:

- Click the dropdown by instances
- Edit port security groups
- Select Edit Security Groups for the interface to add these exceptions to (almost every VM will only have one)
- Add and remove groups using the + and - operators

## 6.17 Is there a default firewall policy on my project (and what does it say ?)

All egress is enabled.

The following ingress is enabled on IPv4 by default:

- All ICMP Traffic
- TCP/22
- UDP/7777

To add additional firewall ports see *How do I access my Virtual machines across the internet?*.

## 6.18 How do I avoid Docker Hub Rate limits?

STFC Cloud has their own docker hub mirror which is available to machines on the internal network. In-depth information can be found at: [STFC Docker Hub Mirror](#).

This is free, does not apply rate limits and faster than Docker Hub, as it pulls over the local network. In the event the mirror is unavailable the instance will automatically pull from Docker Hub Directly.

New instances will use the mirror by default. Existing instances will also receive this change over time, but require restarting the docker service or instance to apply the update (see [Restarting Docker](#)).

If you want to manually verify that an instance is using the Docker Hub mirror see [Checking that the mirror is being used](#).

## 6.19 I have a Database listening on its default port, but I can't connect to it remotely - Why?

Databases are not externally accessible under default policy.

Contact us to discuss the possibility of adding one-off “additonal” policies for specific hosts.

## 6.20 A VM machine I was using has died - How can I get the data off of it?

Depending on the way the VM has failed we may be able to help get this back. Contact us at [cloud-support@stfc.ac.uk](mailto:cloud-support@stfc.ac.uk)

## 6.21 What are the current machine “flavors”? Can I have a machine that looks like a flavor you don't have?

For most people the pre-configured flavors should fit almost every workload. See [Flavors](#) for the types available.

Please contact us if you need assistance creating your own local project flavors.

## 6.22 Can you provide operating system “X” on Openstack?

Policy doc on how we deal with new OS requests

## 6.23 What sort of CPU performance should I expect?

This varies between flavors depending on workload, with c\* flavors generally offering the best performance per core. For a full list of the available flavors see [Flavors](#).

If you have concerns or further queries please feel free to contact us.

## 6.24 What sort of disk I/O performance should I expect locally on a VM?

Instances are currently limited to 200 IOPS read and write.

For reference throughput (note: not latency) is comparable to a 15K SAS disk, or double the speed of a typical hard drive found in a desktop.

## 6.25 What sort of network bandwidth should I expect on a VM?

Hypervisors are currently connected to either 10gb or 25gb links, so you can expect a share of this depending on the size of the VM and the number of VMs on the same host.

## 6.26 Can you recover a VM I accidentally deleted?

Unfortunately, we cannot.

If a volume was attached, and it was not selected during deletion process, it can be attached to a new instance to access the data on it.

## 6.27 How do I login to the “admin” interface?

Visit <https://openstack.stfc.ac.uk>

## 6.28 How do I obtain a host certificate for my Openstack virtual machine?

We don't provide certificates. These can be issued from the internal certificate site, an external vendor or be re-used from an existing deployment.

## 6.29 My host seems to have rebooted since last time I logged in - why?

This is rare, but usually this is due to an issue when migrating a VM which has triggered a reboot.

If you have concerns please feel free to contact us for additional support.

## 6.30 What are the default DNS servers for VMs on Openstack?

By default VMs in OpenStack use the DHCP agents within their Project networks as the DNS server IPs can change. The current DNS server IPs can be viewed by viewing the following file on non-Systemd machines (such as SL7):

```
cat /etc/resolv.conf
```

For Systemd based machines (such as Ubuntu) the current DNS servers can be viewed by doing:

```
systemd-resolve --status
```

## **FAULT FIXES**

### **7.1 How to find and fix VMs in “error” state**

#### **7.1.1 Overview**

How to find the VM’s and possible methods of fixing them

### **7.2 “Neutron is not responding Error” on Openstack Web user interface**

#### **7.2.1 Symptoms**

If you receive an error “Neutron is not responding Error” when listing the instances within a project, or that when looking at the “Networks” in the project, it is not populating the subnets that are part of the Network. On a command line, when running the command “openstack subnet list” the system returns with “An unknown error occurred”. Explanation This issue has been known to occur if a network or subnet or router has been deleted in a particular way, such that the “subnet” still exists, but has been disassociated from the “network”. This means that in the neutron database (on osdb1.nubes.rl.ac.uk), the subnet table has an entry where the “network\_id” IS NULL value. This seems to break the Openstack web user interface.

#### **7.2.2 Solution**

Please contact [cloud-support@stfc.ac.uk](mailto:cloud-support@stfc.ac.uk)

### **7.3 Connectivity https services seem broken from inside a Docker Container**

#### **7.3.1 Symptoms**

Within a docker container, a “curl <http://google.com>” works fine, but a “curl <https://google.com>” would fail. However, outside of the container on the VM natively, both of these will work.

### 7.3.2 Explanation:

This is seen a lot on Cloud hosts where the MTU size is not the usual 1500 bytes, and so it is wise to “clamp” the initiating TCP session for an application to use the same segment size as the MTU of the local host – rather than assuming a possibly invalid (and too large) value.

### 7.3.3 Solution

Since version 17 of Docker, the following entry needs to be added to the iptables policy on the host that is running the containers:-

```
iptables -I FORWARD -p tcp --tcp-flags SYN,text SYN -j TCPMSS --clamp-mss-to-pmtu
```

This sets the “maximum segment size” value of a TCP Syn packet when it initiates a connection as part of “path MTU” negotiation. This is intended to get past the issue of “icmp 3 code 4” packet blocking issues (by the sending of the original packet) where the “Don’t fragment” flag is set on an IP packet, but the packet is too large and the sending router/host does not receive the icmp 3 code 4 packet to change and resend the “too large” packet as a smaller size. This is often an issue when using tunnelling or “packet sleeving” – GRE and IPSEC packets as well.

### 7.3.4 References

<https://www.frozentux.net/iptables-tutorial/chunkyhtml/x4721.html>   <https://stackoverflow.com/questions/47551873/no-http-https-connectivity-inside-docker-container>

## FLAVORS

The STFC Cloud provides a number of flavors to users but these fall into 4 categories at the moment.

No flavor has resilient storage. If this is required you should boot from Volume at VM creation time

Note: our flavors are created in so that they will efficiently pack onto the hardware we have - due to this we cannot create additional flavors as that may lead to scheduling difficulties and under utilisation.

### 8.1 c\*.\* (e.g. c1.small or c3.large)

**Warning:** C-flavors

C flavors are now deprecated and new VMs will not be able to be created after the 12th of April 2023. It is recommend to use l flavors instead.

These are flavors with dedicated CPU cores meaning that a VCPU core assigned to your VM is dedicated to you.

These flavors are great for computation and are optimised to either be within a single NUMA node providing great performance or balanced equally across the NUMA nodes in the hypervisor.

### 8.2 l\*.\*

These are flavors which utilize local SSD storage on the hypervisors with all of the storage being allocated to the root disk. These instances currently cannot be live migrated so in the event of hardware errors the machine will likely be lost or down for an extended period.

Otherwise the flavors behave as c\*.\* flavors

### 8.3 le\*.\*

These are flavors which utilize local SSD storage on the hypervisors with 100GB being allocated to the root disk and the rest to an ephemeral disk. These instances currently cannot be live migrated so in the event of hardware errors the machine will likely be lost or down for an extended period.

Otherwise the flavors behave as c\*.\* flavors

## 8.4 g\*.\*

These have the same CPU configuration as c\*.\* flavors but have a GPU attached:

- g-p4000.\* have one or more NVidia Quadro P4000 cards attached (the smallest has 1, the largest has 4)
- g-rtx4000.\* have one or more NVidia Quadro RTX 4000 cards attached (the smallest has 1, the largest has 4)
- **g-rtx4000-ref.\* have one or more NVidia Quadro RTX 4000 cards attached (the smallest has 1, the largest has 4)**
  - these are old out of warranty nodes which have been refitted with new GPUs. For this reason these are reserved for short term use such as training courses. If you have a use case you want to use these for then please get in touch with support.
- g-a4000.\* have one or more NVidia RTX A4000 cards attached (the smallest has 1, the largest has 4)
- **g-a4000-ref.\* have one or more NVidia RTX A4000 cards attached (the smallest has 1, the largest has 4)**
  - these are old out of warranty nodes which have been refitted with new GPUs. For this reason these are reserved for short term use such as training courses. If you have a use case you want to use these for then please get in touch with support.
- g-p100.\* have one or more NVidia Tesla P100 cards attached (the smallest has 1, the largest has 2)
- g-v100.\* have one or more NVidia Tesla V100 32GB cards attached (small has 1, medium has 2 and large has 4)
- g-a100.\* have one or more NVidia Tesla A100 40GB cards attached (the smallest has 1, the largest has 4)
- **g-a100-n100.\* have one or more NVidia Tesla A100 40GB cards attached (the smallest has 1, the largest has 4) with higher core count and more network bandwidth**
  - These are currently reserved for a specific project.
- g-a100-80gb.\* have one or more NVidia Tesla A100 80GB cards attached (the smallest has 1, the largest has 8)
- **g-arc-a770.\* have one or two Intel Arc A770 16GB cards**
  - These are very limited in number
- **g-w6600.\* have one or more AMD W6600 8GB cards (the smallest has 1, the largest has 4)**
  - These are very limited in number

## IMAGE TYPES

We currently provide several curated images for use with the STFC Cloud.

These are images which have been preconfigured to meet our [Terms Of Service](#) by having automatic updates enabled and reporting in to various monitoring systems.

We currently provide 2 operating systems, Rocky Linux/Scientific Linux (a RedHat derivative) and Ubuntu.

We aim to produce new versions of each image at least monthly with the previous version being renamed and deactivated.

The Images we provide are:

- Rocky-8-Aq (rocky-8-aq)
- Rocky-8-NoGui (rocky-8-nogui)
- ubuntu-focal-20.04-nogui
- ubuntu-focal-20.04-gui
- ScientificLinux-7-AQ (scientificlinux-7-aq)
- ScientificLinux-7-Gui (scientificlinux-7-gui)
- ScientificLinux-7-NoGui (scientificlinux-7-nogui)
- Ubuntu-Bionic-Gui (ubuntu-bionic-18.04-gui)
- Ubuntu-Bionic-NoGui (ubuntu-bionic-18.04-nogui)

### 9.1 Image Variants

We provide 3 different variants of images:

- Gui - Includes a preinstalled desktop environment
- NoGui - A minimal install
- AQ - A minimal install bootstrapped into the Aquilon configuration management system (Only available to Aquilon users within STFC)

## 9.2 Additional Operating systems

We currently only offer the above operating systems as there is an overhead to providing the curated images although we are currently working on the provision of Ubuntu 22.04 and Rocky 9 images.

We do allow users to upload their own images although it is then the users' responsibility to ensure any VMs created with these comply with our *Terms Of Service*. Any non compliant VMs will be deleted.

Please contact us at [cloud-support@stfc.ac.uk](mailto:cloud-support@stfc.ac.uk) if you would like to upload your own images and discuss how to comply with the *Terms Of Service*

## REFERENCE DOCUMENTS

### 10.1 STFC Docker Hub Mirror

#### 10.1.1 Introduction

The STFC cloud team provides an internal Docker Hub mirror at <https://dockerhub.stfc.ac.uk> which is strongly recommended; users avoid rate limit issues, can pull large images faster over the local network and is free.

Docker Hub limit an IP to 100 pulls per 6 hours. Instances with floating IPs will have separate rate limits, whilst all other instances use a shared limit (as all pulls come from a single external IP).

Docker will automatically fall back to using Docker Hub directly if the mirror is ever unavailable, so there is minimal risk to applying these changes.

#### 10.1.2 New Instances

By default all new STFC Cloud instances will use the mirror by default. To confirm the mirror is being used see *Checking that the mirror is being used*

#### 10.1.3 Existing Instances

Existing instances will automatically update to pull in the required file described here *Restarting Docker*. A restart of the instance or docker service is required to start using the mirror.

#### 10.1.4 Checking that the mirror is being used

##### Checking configuration exists

Users can quickly check if they have the mirror configuration present by running:

```
cat /etc/docker/daemon.json
```

An output similar to this indicates that the configuration is present:

```
{  
  "registry-mirrors": ["https://dockerhub.stfc.ac.uk"]  
}
```

To check whether the docker daemon is using this config you can run the following:

```
docker info
```

An output which includes the following near the bottom will indicate that the mirror is being used:

```
Registry Mirrors:
  https://dockerhub.stfc.ac.uk/
```

## Monitoring Logs

For further peace of mind the logs can be checked. Docker will not produce any logs when pulling from the mirror successfully but will log an error if it fails to use direct connections:

- If this an existing instance which has not been restarted recently see [Restarting Docker](#).
- In a terminal follow the Docker logs:

```
# Where -u means unit and -f means follow
journalctl -u docker -f
```

- Select an image which has not been pulled on the System before, for example Ubuntu or Debian latest.
- In a separate terminal run a docker pull of your image:

```
# Assuming ubuntu:latest has never been pulled on the machine
docker pull ubuntu:latest
```

- If the mirror could be contacted there will be no output in the logs about it
- If the mirror could not be used something similar to this will appear:

```
Jan 1 12:00:00 My-Instance-Name dockerd[900]: time="2021-01-01T12:00:00.000000000Z"
↳level=info msg="Attempting next endpoint for pull after error: ..."
```

For more support see [Troubleshooting Mirror Connection](#).

## 10.1.5 Manually Configuring the Mirror

For most users this is not required; instances will automatically update the file as described in [Checking that the mirror is being used](#).

If you have internal machines that are outside of Openstack or separately managed or you need to apply the changes proactively the following steps can be followed:

### Docker Daemon

If you are using Docker (SL7 / Ubuntu / CoreOS / K8s < 1.20) the following steps can be performed using *sudo* or *root*. By default most distributions do not pre-create this file:

```
mkdir -p /etc/docker
<editor> /etc/docker/daemon.json # e.g. vi /etc/docker/daemon.json
```

Add or append the following JSON:

```
{
  "registry-mirrors": ["https://dockerhub.stfc.ac.uk"]
}
```

Then restart the service (see [Restarting Docker](#)).

## Containerd

As of Kubernetes 1.20, a future release (TBA) will use Containerd by default. The STFC Core OS image already contains the mirror information on users behalf at `/etc/containers/registries.conf`

Further documentation on manually on setting up containerd will be included when upstream Kubernetes uses containerd by default and further internal testing is completed.

### 10.1.6 Restarting Docker

**Warning:** Restarting the Docker Daemon will also pause and resume any running containers. This may result in service interruption or lost work.

The Docker Daemon can be restarted to apply the changes. Any running containers will be either paused and resumed, or restarted during this process.

Depending on the operating system the service is either called *docker* or *dockerd*, with the former being more common. To restart simply run:

```
systemctl restart docker # or dockerd
```

To verify the service has resumed successfully:

```
systemctl status docker # or dockerd
```

### 10.1.7 Troubleshooting Mirror Connection

The logs can be checked if you suspect the mirror is not being used (see [Monitoring Logs](#)).

If Docker is failing back some simple diagnostics steps can be performed and/or cloud support can be contacted. Including the output of these steps in your email will enable us to provide support faster:

- Check that the VM has a connection to the internet with `ping example.com`
- Check config matches [Manually Configuring the Mirror](#), paying attention to typos in the URL
- Check that you can connect to the mirror using curl: `curl https://dockerhub.stfc.ac.uk/v2/` you should get an empty response: `{}`
- Check the systemd logs for additional errors related to Docker `journalctl -u docker`
- Contact us for additional support

## 10.2 Jupyter Hub Service

### Contents

- *Jupyter Hub Service*
  - *Introduction*
    - \* *Advantages*
  - *Resources Available*
    - \* *Storage*
    - \* *Lifetime*
    - \* *Available Instances*
      - *Default Instance*
      - *Large Memory Instance*
      - *GPU Instance*
    - \* *Switching Instances*
  - *Logging in with IRIS IAM*
    - \* *Account Creation*
    - \* *Group Membership*
      - *As a individual*
      - *As a group administrator/owner*
  - *Out of Memory Killer*
    - \* *Lowering Memory usage*
  - *Troubleshooting / FAQ*
    - \* *“I cant launch a notebook”*
    - \* *“My notebook randomly stops processing”*
    - \* *“I have a problem with my notebook...”*
  - *Running Own Service*

### 10.2.1 Introduction

**Warning:** This is currently in a testing phase, you may experience issues or data loss and the service is currently provided-as-is with no guarantees.

Please regularly backup your work as Jupyterhub storage may be reset during this phase.

STFC Cloud provides a JupyterHub service for IRIS IAM select group members. This service is available both internally and externally (i.e. without a VPN).

During testing we are primarily to gauging interest and technical feasibility. If you run into any problems, or have any feedback please forward it on using either Slack or the ticketing system ([Contact Details](#))

## Advantages

Using the JupyterHub service provides a number of advantages, including:

- Access to modern server processing, speeding up reductions
- Ability to work without using remote-access onto a desktop
- Ability to switch devices and resume work both on-site and off-site
- Users without access to powerful compute can process data

After gauging interesting and completing testing we may offer the following further advantages (subject to change):

- Instances with a large number of CPU cores
- Instances with large memory allocations
- Instances with Nvidia GPU Access

## 10.2.2 Resources Available

As a result of recent limited site access and a hardware installation backlog we are currently providing small Jupyterhub instances.

During testing we plan on measuring node usage to plan future instance sizing.

## Storage

**Warning:** This is not currently backed up and can be lost on service updates/upgrades or unplanned cluster outage. Please keep a copy of any important data locally.

All instances come with a 1GB data store, which is carried between all types. This should be viewed as a working scratch space rather than somewhere to store data long-term.

## Lifetime

During our testing phase the lifetime of an instance is 1 day of idle time. After this time period the instance will be stopped, but storage will be retained. On next login the user will be prompted to restart the service.

## Available Instances

### Default Instance

The default instance comes with the following resources, it's planned to cover most users requirements:

- CPU: 2
- RAM: 1.5Gb

## Large Memory Instance

This “large” memory instance comes with a larger memory and CPU allowance. It’s planned to cover Jupyter scripts which process large datasets.

- CPU: 8
- RAM: 7.5GB

## GPU Instance

This is **not** available currently due to pending technical work.

This would provide a dedicated Nvidia GPU instance for users with CUDA/OpenGL workflows.

## Switching Instances

To switch between available Instances the original instance must be shutdown:

- Complete / Save any work on the existing instance
- Go to the top menu-bar -> **File** -> **Hub Control Panel**
- In the new tab select **Stop My Server**
- You will be returned to the instance selection screen
- Any existing storage is automatically transferred between instances

## 10.2.3 Logging in with IRIS IAM

### Account Creation

A IRIS IAM account is required. Help for registering with IRIS IAM can be found *here*. <<https://iris-iam.stfc.ac.uk/help/>>

## Group Membership

### As a individual

- Sign into [IRIS](#)
- Select *View Profile Information*
- The list of groups you belong to is in the top-right, by default this will be empty.
- Under **Group Requests** select, **Join a group**.
- Enter the group name you wish to join and submit. You will need to wait for your group administrator(s) to approve your request.
- When the request is accepted the group name will appear in the group list
- If you are still unable to access the service contact your group administrator/owner to verify if the group has requested access to Jupyterhub.

## As a group administrator/owner

If you'd like to add access to an entire IRIS group please [Contact Support](#) with the following, to help us plan capacity:

- The IRIS IAM Group name
- Estimated number of users
- Estimated instance sizes (e.g. we typically use xGB RAM)

### 10.2.4 Out of Memory Killer

The Out of Memory (OOM) Killer will fire when Python tries to allocate memory, but the instance has run out.

Unfortunately, there is no warning or error, as we require memory (which has run out) to send the message that were out of memory. From a users perspective it will look something like this:

- Despite using run all, we have only partially completed the notebook
- There will be no \* or numeric character next to the cell where we ran out of memory
- The Kernel busy indicator (grey circle in top left) will remain busy.

Simply re-running the script will restart Python automatically.

### Lowering Memory usage

Unfortunately we are not able to support Python directly, but here are some pointers to help resolve your issue:

- Notebooks share memory on an instance once they have been run. These can be shutdown from the top menu-bar -> **Kernel** -> **Shut Down (All) Kernel(s)**
- Consider the size of the dataset, 100MB of raw data should not saturate a 2GB instance. Equally 10GB of data will require a 16GB instance.
- Check for unnecessary copies and duplicated data, for example

```
raw_data = load_numpy_data()
to_process = raw_data.copy()
processed = my_library.some_operation(to_process)
```

Depending on how some operation works we could either have 2x or 3x the original data size. Instead we should set these to None after were done to allow Python to clear them. (Note: avoid *del*, as this could cause bugs later in your program with no added benefit over setting it to *None*)

```
raw_data = load_numpy_data()
# Copy was removed and raw_data was passed straight in
processed = my_library.some_operation(raw_data)

# Now we allow Python to garbage collect raw_data
raw_data = None
```

- Advanced users might want to use [a memory profiler](#).

## 10.2.5 Troubleshooting / FAQ

### “I cant launch a notebook”

Notebooks typically take 1-2 minutes to bring your storage online.

If multiple users create instances at roughly the same time it will exhaust our limited number of placeholders and will take the service 10-15 minutes to scale to the new capacity. This will result in a timeout error when creating your service, please retry after 20 minutes.

If you continue to receive timeouts please contact us ( [Contact Details](#)) with details of the error you are receiving.

### “My notebook randomly stops processing”

- Check that your notebook is not being killed by the *Out of Memory Killer*
- Check that your notebook hasn't entered an infinite loop
- Check the Python a library version match your notebook requirements, you may need to pin library versions.
- If possible, try testing your notebook locally completes:
- If your notebook completes locally please contact us for support
- If your notebook fails locally please contact your notebook provider

### “I have a problem with my notebook...”

If you suspect there is a problem with your instance please contact us ([Contact Details](#)).

Unfortunately the cloud team is unable to provide support for Python problems. Please contact your notebook provider for additional support or consult the [Jupyterhub Documentation](#)

## 10.2.6 Running Own Service

Some understanding of Openstack and Linux CLI usage is required to deploy an instance. User-facing configuration, such as worker flavor, profile sizes / descriptions and GPU support, is written in .yaml files so knowledge of Ansible is not required.

The service is packaged and built specifically for *jupyter.stfc.ac.uk* out of the box. Variables to control these are split into two files; one for Openstack (number/size of instances...etc.) and one for Jupyterhub (domain name / auth method...etc.).

This allows anybody with access to a Openstack environment with Magnum to run a similar service on any domain.

For requirements, deployment instructions and available features/limitations see the [STFC Ansible Jupyter Repository](#).

## 10.3 Harbor Private Docker Registry

### Contents

- *Harbor Private Docker Registry*
  - *Background*
  - *Getting Started*
    - \* *Signing Up*
    - \* *Signing In*
    - \* *Creating a Project*
  - *Using Harbor*
    - \* *Project Permissions*
    - \* *Private Projects*
    - \* *Vulnerability scanner*
    - \* *Authenticating Docker*
    - \* *Rotating Access Token*
    - \* *Tagging and Pushing Images*
    - \* *Advanced: Project Retention Rules*

### 10.3.1 Background

The STFC Cloud team runs a Harbor private registry at <https://harbor.stfc.ac.uk>

This service allows users to upload container images, and manage their lifecycles internally. This has a number of advantages over using a public repository:

- Login uses IRIS IAM, so existing credentials can be used rather than creating new accounts and teams on external providers.
- Images aren't subjected to free / paid account restrictions on other platforms.
- Built in vulnerability scanner indicates any CVEs applicable to the image on upload and a periodic basis.
- Projects can be made public, or privately available to a specific group of users.

### 10.3.2 Getting Started

#### Signing Up

Anybody with an IRIS account is able to sign up. An IRIS account can be applied for using facility credentials [here](#).

To associate your IRIS account simply go to [Harbor](#) and click **Login Via OIDC Provider**. If it's your first sign in, you will be asked to confirm basic details being shared with Harbor. Harbor will then confirm the account details (which cannot be modified).

Help for registering with IRIS IAM can be found [here](https://iris-iam.stfc.ac.uk/help/). <<https://iris-iam.stfc.ac.uk/help/>>

## Signing In

If you have already signed up (as described in [Signing Up](#)) you can login by simply clicking **Login Via OIDC Provider**. Entering your facility/IRIS credentials into the Harbor login page will not work, as IRIS IAM handles authentication on our behalf.

## Creating a Project

Please [contact us](#) to request a new project, please include the following details in your email:

- The project name (all lowercase, can use - and \_)
- Brief description of planned usage
- Is the project public or private (see [Private Projects](#))
- Estimated storage requirements (GB)
- Planned User / Group Permissions (see [Project Permissions](#))
- (Optional) Retention Rules (see [Advanced: Project Retention Rules](#))

## 10.3.3 Using Harbor

### Project Permissions

Permissions are presented in increasing scope, each level presented includes new permissions and all permissions from the previous level:

- **Anonymous**  
A user who has not logged into Harbor using their Docker client. They can only pull from public repositories.
- **Limited Guest**  
A user who can pull from private and public projects, but cannot see logs or members of a project. Useful for end-users and machines.
- **Guest**  
A user who can pull images from a private project and see other members of the project.
- **Developer**  
A user who can pull and push images to the project. They cannot delete an image once pushed.
- **Project Master**  
A user who can also trigger manual security scans and delete images.
- **Project Admin**  
This person can manage all aspects of a named project, including changing user and group permissions.
- **Harbor Administrators**  
Can create and delete projects. Can update a project's admin to be a different user as required upon support request.

## Private Projects

### Important: Secrets in Images

A private project/repository does not mean users should include secrets into their Docker images. Please keep secrets separate to images by passing them through .env files or environment flags. A good rule-of-thumb is asking, “if this image ever leaked could a system become compromised from the details within”.

Users can request a private repository; the names, images and associated members of these projects are hidden from non-members.

For most use-cases a public project is preferred:

- Images are immutable; a SHA reference cannot be changed.
- Anonymous pulls removes the requirements on securely distributing and storing access tokens.
- Users can start software with a single docker command, lowering the barrier of entry for deployment.

Some examples where a private projects should be considered are:

- When software licenses are required per container instance
- Mirroring / storing proprietary software (check License Agreement beforehand)
- Confidential or unannounced/internal development projects
- Where scientific data is included but subject to access restrictions

Machines will not be able to pull from a private repository without first *logging in*.

## Vulnerability scanner

By default we configure all projects to automatically scan images pushed to harbor for vulnerabilities and we also conduct weekly scans against all images.

You can view the results by clicking into the project. Then into the repository and then you will see a summary of the vulnerabilities against each artifacts

You can hover over the Vulnerabilities field to view a summary or click into the artifact to view further details

We recommend that you resolve all Critical and High rated vulnerabilities as soon as possible.

Project Administrators can configure the project to disallow pulling of images based on the vulnerabilities against the image.

## Authenticating Docker

**Warning:** A *credentials store* is highly recommended. On machines without a credentials store your token is stored in plain-text within your user profile.

Logging in grants you the ability to pull and push to projects where you have appropriate permissions:

- *Sign into Harbor*
- Take note of your profile name in the top-right
- Click on the profile name and click **User Profile**

- Copy the CLI secret can be copied using the copy action
- On the target machine run

```
docker login -u <profile_name> https://harbor.stfc.ac.uk
```

- It will prompt you for your access token, paste in the previously copied token
- Docker will return if the login was a success and persist this between reboots

## Rotating Access Token

This is useful if your Docker token has been, or is possibly compromised, or on a machine you no longer have access to. Rotating keys does **not** flag or log your account in any way, so please feel free to use this proactively.

Rotating the access token will generate a new token whilst invalidating the old token and is simple:

- *Sign into Harbor*
- Click on the profile name and click **User Profile**
- Click the 3 dots next to **CLI secret**
- Select **Generate Secret**
- Confirm you are happy to discard your old token
- On each machine you require access *re-login*

## Tagging and Pushing Images

Images should include the name of the harbor server, or they will implicitly use Docker Hub:

```
# For tagging as part of the build
docker image build . -t harbor.stfc.ac.uk/<project_name>/<image_name>:<tag>

# For re-tagging an existing image
docker tag <old_tag> harbor.stfc.ac.uk/<project_name>/<image_name>:<tag>
```

Here is a worked example using the image *Ubuntu*, on the *latest* tag to a project called *harbor\_example*

```
# Build a new Ubuntu image
docker image build ubuntu -t harbor.stfc.ac.uk/harbor_example/ubuntu:latest

# For re-tagging an existing Ubuntu image
docker tag ubuntu/ubuntu:latest harbor.stfc.ac.uk/harbor_example/ubuntu:latest
```

To push an image to the repository the following command can be used:

```
docker push harbor.stfc.ac.uk/<project_name>/<image_name>:<tag>
```

For example to mirror the image *ubuntu:latest* from Docker Hub into Harbor Project *my\_project*:

```
# This assumes the tag step above was completed
docker image push harbor.stfc.ac.uk/my_project/ubuntu:latest
```

## Advanced: Project Retention Rules

### Requests

Up to 15 retention rules can be set on a per-project basis.

Harbor will consider all repositories and all tags eligible for deletion after a user specified number of day **or** after a number of artifacts.

We can also white-list or black-list tag patterns or repository names that are subject to auto-retention rules.

For example, in a project with repositories *foo*, *bar* and *baz* we can specify only *foo*, *baz* to be auto collected after 60 days, whilst *bar* will only delete tags with *\*beta\** in their name after 20 days.

If your putting in a support request for retention rules please describe the above in a request, we will configure the rules per your description. For users manually confusing their rules an additional reference follows.

### Manual Config

Harbor currently has limited regex capabilities for expressing rules. By default the repository list and tags are set to everything *\*\**.

To specify a list of items, for example *foo bar baz* replace *\*\** with *{foo,bar,baz}* which will match all.

Care must be taken with semantic versioning. Unlike regex a *\** character will only match a single character input, for example a retention rule for *v\*.\*-beta* will match *v1.1-beta* and *v9.5-beta* but not *v10.1-beta*. Full regex support is currently in the feature-request stage upstream.

The rules for a project can be configured by:

- Navigating to the project
- Select the Policy tag
- Ensure Tag Retention is selected
- Configure the rules and schedule as required

## 10.4 Python SDK

### Contents

- *Python SDK*
  - *Introduction*
  - *Setting Up Clouds.yaml*
    - \* *Background*
    - \* *Setup*
    - \* *Accessing Multiple Projects*
    - \* *Auth in Python Script*
  - *Using the Openstack SDK*
    - \* *Documentation*
    - *Worked Example - Set Server Metadata*

### 10.4.1 Introduction

This document provides a brief setup guide for advanced users who want to use the Python SDK. This document assumes a good understanding of Python.

### 10.4.2 Setting Up Clouds.yaml

#### Background

Clouds.yaml allows a user to provide their credentials for a project separate to their script. This is strongly recommended for a number of reasons:

- Prevents hard-coded credentials leaking in the script
- Allows sharing of scripts for support reasons and to colleagues
- Able to quickly switch between a testing and production project

#### Setup

- Navigate to [Openstack](#)
- Go to **API Access**
- On the right click the dropdown to download the *clouds.yaml* file, not the RC file
- Move this file to `~/config/openstack` you may need to create the parent directory

Your YAML file will look similar to:

```
clouds:
  openstack:
    auth:
      auth_url: https://openstack.stfc.ac.uk:5000/v3
      username: ...
```

- Modify the project name (default *openstack*) to something descriptive
- Add your password under the auth directive

For the above example:

```
clouds:
  my_proj_name:
    auth:
      auth_url: https://openstack.stfc.ac.uk:5000/v3
      username: exampleUser
      password: examplePassword
```

## Accessing Multiple Projects

This process can be repeated if you require access to multiple projects.

- Switch project and download YAML for additional project
- Append the contents of this downloaded file under the *clouds* directive

Here is an example of a file with multiple projects:

```
clouds:
  my_proj_name:
    auth:
      auth_url: https://openstack.stfc.ac.uk:5000/v3
      username: ...
      password: ...
  proj_dev:
    auth:
      auth_url: https://openstack.stfc.ac.uk:5000/v3
      username: ...
      password: ...
  proj_prod:
    auth:
      auth_url: https://openstack.stfc.ac.uk:5000/v3
      username: ...
      password: ...
```

This provides targets called *proj\_one*, *proj\_dev* and *proj\_prod*. This is especially useful for testing a script against a development environment and performing a one-line change to switch to production.

## Auth in Python Script

Openstack will automatically go to `~/config/openstack/` to check for a `clouds.yaml` file (note: lower-case).

Using the above examples, we can easily point to the project named *my\_proj\_name* like so:

```
import openstack

conn = openstack.connect(cloud='my_proj_name', region_name='RegionOne')

# Print the list of servers to prove were connected as expected
s = conn.compute.servers()
print(s)
```

### 10.4.3 Using the Openstack SDK

**Important:** Because the Openstack SDK dynamically populates the connection object with available methods most IDE type suggestions will be internal low-level methods and should be ignored.

For example *set\_metadata* is always available instead of *set\_x\_metadata*, where x is a Openstack service.

The Openstack SDK dynamically populates the connection object at runtime. This is because Openstack fundamentally is a modular install with different plugins enabled by operators.

Instead Openstack adds “Proxies” to a connection object to represent what is available on the service. The list of proxies can be printed at runtime with:

```
import openstack

connection = openstack.connect(cloud='my_proj_name', region_name='RegionOne')
print(dir(connection))
```

## Documentation

The full list of [proxies](#) can be viewed [here](#).

## Worked Example - Set Server Metadata

The [compute proxy](#) allows us to manipulate compute resources (i.e. instances).

Looking at the documentation linked above there are methods to get a list of server instances and a method to set metadata. If we want to set an arbitrary field of *FOO=BAR* on all servers we can do so using the compute module:

```
import openstack

connection = openstack.connect(cloud='my_proj_name', region_name='RegionOne')
server_list = connection.compute.servers()
for server in server_list:
    connection.compute.set_server_metadata(server, FOO='BAR')
```

## 10.5 Network Bandwidth Expectations

### 10.5.1 What sort of network performance should I expect?

The network bandwidth available to VMs will depend on the type of network your VMs are on and the contention on the hypervisor on which your VM is running:

Internal (172.16.\*.\*):

Private (project private networks eg 10.\*.\*.\* or 192.168.\*.\*):

### 10.5.2 How can I test network bandwidth?

The best way to test network bandwidth is with `iperf3`. You will need two hosts, the location of these hosts will vary depending on what you are trying to test. If you want to test connection to an STFC service please contact us and we can arrange one of the hosts for testing.

On both hosts install `iperf3`

```
yum install iperf3 -y
```

On one host (the server) start `iperf3` in server mode (if we are providing a host to test with skip this)

```
iperf3 -s
```

On the second host (the client) start

```
iperf3 -c <ip of your first machine>
```

On the client side you should see an output like this:

```
[root@test-private-network-3-1 ~]# iperf3 -c 192.168.248.65
Connecting to host 192.168.248.65, port 5201
[ 4] local 192.168.248.52 port 51932 connected to 192.168.248.65 port 5201
[ ID] Interval           Transfer     Bandwidth       Retr   Cwnd
[ 4]  0.00-1.00    sec   1.87 GBytes  16.0 Gbits/sec    75   1.66 MBytes
[ 4]  1.00-2.00    sec   2.10 GBytes  18.0 Gbits/sec   159   1.39 MBytes
[ 4]  2.00-3.00    sec   2.13 GBytes  18.3 Gbits/sec    53   1.77 MBytes
[ 4]  3.00-4.00    sec   2.11 GBytes  18.2 Gbits/sec   256   1.45 MBytes
[ 4]  4.00-5.00    sec   2.14 GBytes  18.4 Gbits/sec    32   1.95 MBytes
[ 4]  5.00-6.00    sec   2.16 GBytes  18.5 Gbits/sec   209   1.76 MBytes
[ 4]  6.00-7.00    sec   2.16 GBytes  18.6 Gbits/sec     6   1.51 MBytes
[ 4]  7.00-8.00    sec   2.14 GBytes  18.4 Gbits/sec   110   1.56 MBytes
[ 4]  8.00-9.00    sec   2.15 GBytes  18.5 Gbits/sec    46   1.86 MBytes
[ 4]  9.00-10.00   sec   2.16 GBytes  18.5 Gbits/sec    42   1.69 MBytes
- - - - -
[ ID] Interval           Transfer     Bandwidth       Retr
[ 4]  0.00-10.00   sec   21.1 GBytes  18.1 Gbits/sec   988
[ 4]  0.00-10.00   sec   21.1 GBytes  18.1 Gbits/sec
                                sender
                                receiver

iperf Done.
```

From this you can see that there was an average bandwidth of 18.1 gbps during this test

## 10.6 Lifecycle of a Virtual Machine

Managing VMs is key to maintaining good security. Preparing a plan to patch, reboot and migrate to new OSes is the best way to ensure maintenance is carried out regularly.

The general attitude towards VMs should be that they are cattle, not pets - any services/workflows running on a VM should be easily replicable so that an individual host is not required to be kept around indefinitely. There are many ways to simplify setting up a VM, such as [Magnum clusters](#) and [Heat stacks](#).

A good lifecycle management process also promotes good service management practices.

### 10.6.1 VM age

VMs should not be left to age for too long for a few reasons, primarily security - vulnerabilities are more common in older machines, not all are patched. Also the hardware underneath and the flavor of the VM are not immortal - at some point they will no longer work/be supported.

We recommend getting rid of VMs at an age of 6 months if possible, and older than a year is usually not ideal. If your machines are approaching this age, consider migrating to newer flavors.

## 10.6.2 Steps to take

As a user of the STFC Cloud, you are responsible for your machines, whilst the Cloud team is responsible for the OS images (excluding custom images). Some steps you can take to keep your machines in good condition are:

- Cycling out VMs on a schedule (ideally once every 6 months)
- Update VMs regularly, and reboot them (many updates do not take effect until after a reboot)
- Comply with security notifications from the Cloud team

Patching and rebooting VMs will extend their lifecycle by a certain amount, but eventually all will need to be replaced.

## 10.6.3 Making things easier

- Configuration management is your friend - can create machines with packages pre-installed
- If you're running a service - design for high availability so that one machine going down for maintenance doesn't take the service down with it
- For one-off VMs - use [Openstack Volumes](#) or other non-root-disk storage so that data isn't lost when cycling VMs
- Stick to a reboot schedule

Finally, the Cloud team are here to help - submit a ticket if you have issues or use the Slack to ask questions.

## AODH AND GNOCCHI

### 11.1 Create Aodh Alarms

**Warning:** This component is currently disabled and not available.

Aodh alarms can be created using the Openstack CLI or by using Aodh Resources in a heat template.

#### 11.1.1 Overview

Alarms provide monitoring-as-a-service for user's resources running on Openstack. One application of these alarms is to create autoscaling stacks, where alarms determine whether a scale-up or scale-down policy is applied to a group of instances.

Gnocchi metrics can be used to create threshold alarms, for example.

The Aodh and gnocchi CLIs can be installed using:

```
pip install aodhclient
pip install gnocchiclient
```

#### Types of alarms

There are three different kinds of alarms:

- **Threshold:** Alarms are triggered when a threshold have been met.
  - Valid Threshold Alarms:
    - \* gnocchi\_resources\_threshold
    - \* gnocchi\_aggregation\_by\_metrics\_threshold
    - \* gnocchi\_aggregation\_by\_resources\_threshold
- **Composite Alarms:** Alarms which have been defined with multiple triggering conditions.
- **Event Alarms:** Alarms when an event has occurred, such as an instance being powered down or in error state.

Alarms can have three different states:

- **ok:** The alarm rule has been defined as False
- **alarm:** The alarm rule has been defined as True

- **insufficient data:** There are not enough data points in order to determine the state of the alarm.

### 11.1.2 Alarm Commands

```

openstack alarm list #lists user-created alarms

openstack alarm create <options> # create an alarm

openstack alarm update <options> <alarm-id> #update the properties of an alarm

openstack delete <alarm-id> #delete an alarm

openstack alarm show <alarm-id> #show details of an alarm

openstack alarm-history show <alarm-id> #see history of the alarm

openstack alarm-history search #search history for all alarms based on query

openstack alarm state set #set the state of an alarm

openstack alarm state get #get state of an alarm

```

To list user-created alarms, use the command:

```
openstack alarm list
```

This returns an empty line if no alarms have been created, or returns a list of alarms such as:

```

+-----+-----+-----+-----+
| alarm_id | type | name |
+-----+-----+-----+
| state | severity | enabled |
+-----+-----+-----+
| alarm-id-1 | event | instance_off |
| alarm | low | True |
| alarm-id-2 | gnocchi_resources_threshold | memory_use_test |
| ok | low | True |
+-----+-----+-----+

```

#### Create an Alarm

To create an alarm, use the command:

```
openstack alarm create <options>
```

### 11.1.3 Create Alarm Options

```
openstack alarm create [-h] [-f {json,shell,table,value,yaml}]
                        [-c COLUMN] [--noindent] [--prefix PREFIX]
                        [--max-width <integer>] [--fit-width]
                        [--print-empty] --name <NAME> -t <TYPE>
                        [--project-id <PROJECT_ID>]
                        [--user-id <USER_ID>]
                        [--description <DESCRIPTION>] [--state <STATE>]
                        [--severity <SEVERITY>] [--enabled {True|False}]
                        [--alarm-action <Webhook URL>]
                        [--ok-action <Webhook URL>]
                        [--insufficient-data-action <Webhook URL>]
                        [--time-constraint <Time Constraint>]
                        [--repeat-actions {True|False}]
                        [--query <QUERY>]
                        [--comparison-operator <OPERATOR>]
                        [--evaluation-periods <EVAL_PERIODS>]
                        [--threshold <THRESHOLD>]
                        [--event-type <EVENT_TYPE>] [-m <METER NAME>]
                        [--period <PERIOD>] [--statistic <STATISTIC>]
                        [--granularity <GRANULARITY>]
                        [--aggregation-method <AGGR_METHOD>]
                        [--metric <METRIC>]
                        [--resource-type <RESOURCE_TYPE>]
                        [--resource-id <RESOURCE_ID>]
                        [--composite-rule <COMPOSITE_RULE>]
                        [--stack-id <STACK_NAME_OR_ID>]
                        [--pool-id <LOADBALANCER_POOL_NAME_OR_ID>]
                        [--autoscaling-group-id <AUTOSCALING_GROUP_NAME_OR_ID>]
```

**–name:** Alarm name - this should be unique to the alarm in the project.

**–type:** Type of alarm - event, composite, threshold, gnocchi\_resources\_threshold, gnocchi\_aggregation\_by\_metrics\_threshold, gnocchi\_aggregation\_by\_resources\_threshold, loadbalancer\_member\_health.

**–description:** Free text description of the alarm.

**–state:** State of the alarm - ok, alarm, insufficient data.

**–severity:** Severity of the alarm - low, moderate, critical.

**–enabled:** Determine whether the alarm is evaluated. True if alarm evaluation is enabled.

#### Actions when alarm changes state:

**–alarm-action:** [Webhook URL] URL to invoke when alarm transitions to alarm state.

**–ok-action:** [Webhook URL] URL to invoke when alarm transitions to ok state.

**–insufficient-data-action:** [Webhook URL] URL to invoke when alarm transitions to insufficient data state.

These actions may be used multiple times.

**–time-constraint:** Only evaluate the alarm if it is within the time constraint. Start point(s) specified with a cron expression, and duration in seconds. Can be specified multiple times for multiple constraints. Format: `:bash: name=<CONSTRAINT_NAME>;start=<CRON>;duration=<SECONDS>;[description=<DESCRIPTION>;[timezone=<IANA Timezone>]]`

**–repeat-actions:** [Default to False] Determines whether actions should be repeatedly notified why the alarm remains in target state.

### Alarm Rules

#### –query:

- Threshold or Event Type: key[op]data\_type::value; list. data\_type is optional, but if supplied must be string, integer, float, or boolean
- gnocchi\_aggregation\_by\_resources\_threshold: need to specify a complex query json string, like: :bash: `{“and”: [{“=”: {“ended_at”: null}}, ...]}`

**–comparison-operator:** Operator to compare with - lt, le, eq, ne, ge, gt.

**–evaluation\_periods:** Number of periods to evaluate over.

**–threshold:** Threshold to evaluate against.

### Event Alarm

**–event-type:** Event type to evaluate against.

### Threshold Alarm

**–m, –meter-name:** Meter to evaluate against.

**–period:** length of each period (seconds) to evaluate over.

**–statistic:** Statistic to evaluate - max, min, avg, sum, count.

### Common gnocchi alarm rules

**–aggregation-method:** The aggregation\_method to compare to the threshold.

**–metric, –metrics:** The metric id or name depending of the alarm type

### Gnocchi resource threshold alarm:

**–resource-type:** The type of resource.

**–resource-id:** The id of a resource.

### Composite Alarm:

**–composite-rule:** Composite threshold rule with JSON format, the form can be a nested dict which combine threshold/gnocchi rules by “and”, “or”. For example, the form is like: `{“or”:[RULE1, RULE2, {“and”: [RULE3, RULE4]}]}`, The RULEx can be basic threshold rules but must include a “type” field, like this: `{“threshold”: 0.8, “meter_name”: “cpu_util”, “type”: “threshold”}`

### Loadbalancer member health alarm

**–stack-id:** Name or ID of the root / top level Heat stack containing the loadbalancer pool and members. An update will be triggered on the root Stack if an unhealthy member is detected in the loadbalancer pool.

**–pool-id:** Name or ID of the loadbalancer pool for which the health of each member will be evaluated.

**–autoscaling-group-id:** ID of the Heat autoscaling group that contains the loadbalancer members. Unhealthy members will be marked as such before an update is triggered on the root stack.

After the alarm is created, you should get the alarm details similar to this one:

+-----+-----+	
↪-----+	
Field	Value
↪	

(continues on next page)

(continued from previous page)

+-----+		
↪	alarm_actions	[]
↪		
↪	alarm_id	ALARM_ID
↪		
↪	description	Instance powered OFF
↪		
↪	enabled	True
↪		
↪	event_type	compute.instance.power_off.*
↪		
↪	insufficient_data_actions	[]
↪		
↪	name	power_off_alarm
↪		
↪	ok_actions	[]
↪		
↪	project_id	PROJECT_ID
↪		
↪	query	traits.instance_id = INSTANCE_ID
↪		
↪	repeat_actions	False
↪		
↪	severity	low
↪		
↪	state	insufficient data
↪		
↪	state_reason	Not evaluated yet
↪		
↪	state_timestamp	YYYY-MM-DDTHH:MM:SS
↪		
↪	time_constraints	[]
↪		
↪	timestamp	YYYY-MM-DDTHH:MM:SS
↪		
↪	type	event
↪		
↪	user_id	USER_ID
↪		
+-----+		
↪	-----	

## 11.1.4 Alarms: Threshold, Event, Composite

### 11.1.5 Threshold Alarm

Threshold alarms will change to alarm state when a threshold has been met.

The threshold is based of the value which is returned by the metric.

The metrics for instances are:

Metric	Unit
cpu	ns
disk.ephemeral.size	GB
disk.root.size	GB
memory.usage	MB
memory	MB
vcpus	vcpu
compute.instance.booting.time	sec

These metrics can be used when defining a gnocchi threshold alarm.

**Note:** The alarm granularity must match the granularities of the metric configured in Gnocchi, otherwise the alarm will only return an ‘insufficient data’ state.

**Granularity:** This refers to the time interval (in seconds) in which data is collected.

The following example shows how a threshold alarm can be created for an instance. This alarm is triggered when the memory usage exceeds 10%. However, as the memory usage meter only measures the number of MBs of memory being used.

In order to attach a 10% memory usage alarm to an instance which has 1GB RAM, the threshold is simply 102 MB (which is approximately 10% of 1024MB).

```
openstack alarm create --name memory_use_test
--type gnocchi_resources_threshold
--description "A test alarm which alarms when the memory usage exceeds 10% of RAM. For
↳ this VM, this would be 102MB."
--metric memory.usage
--threshold 102
--comparison-operator gt
--aggregation-method mean
--granularity 300
--evaluation-periods 2
--resource-id <resource-id>
--resource-type instance
```

So if, in two five minute evaluation periods the average memory usage is greater than 102MB, the alarm state will transition to alarm. If the memory usage is below 102MB, the alarm state remains in the ok state.

**Q:** Can CPU utilization alarms be created?

**A:** Unfortunately, the *cpu\_util* meter has been deprecated since the Stein release and so it is not possible to create alarms which monitor the CPU utilization of instances.

**Q:** What about the *cpu* meter? Can the data from that meter be used and converted into % for CPU utilization?

**A:** Although the CPU is monitored it is measured in ns and the *openstack create alarm* command does not allow operations to be performed on the meter in the `--meter` option. However, CPU utilization can be calculated manually using the *gnocchi* command:

```
gnocchi aggregates '(* (/ (aggregate rate:mean (metric cpu mean)) 3000000000000) 100)' \
--id=INSTANCE_ID
```

### 11.1.6 Event Alarm

The following example is an event alarm which transitions to ‘alarm state’ when an instance has no power (or has been powered off).

```
openstack alarm create --type event \
--name instance_off \
--description 'Instance powered OFF' \
--event-type "compute.instance.power_off.*" \ #event to monitor
--enable True
--query "traits.instance_id=string::INSTANCE_ID"
```

**Q:** The event alarm seems to be stuck in ‘insufficient data’ state and states that it has not been evaluated yet.

**A:** Unlike threshold alarms, event alarms will only change state when a specific event has occurred. This means that event alarms will only transition to an alarm state. This also means that event alarms do not transition to ok state either.

**Q:** When an event has happened and the alarm has fired, will the alarm reset?

**A: Event alarms are not reset automatically, so when they are in alarm state, they will stop being evaluated.**

For a power alarm on an instance, once the instance has been powered on the alarm state will need to be changed manually in order for the alarm to be evaluated again. When the state has been changed to ‘insufficient data’ or ‘ok’ using *openstack alarm update* or *openstack alarm state set*, the alarm will be monitored again and the alarm will move to ‘alarm’ state if the event occurs again.

### 11.1.7 Composite Alarms

Composite alarms use a combination of defined rules to determine the state of an alarm. These can be a combination of rules about the threshold of metrics being exceeded, or whether a combination of events has occurred.

```
openstack alarm create
--name composite-alarm-test \
--type composite \
--composite-rule '{"or": [{"threshold": 500, "metric": "memory.usage", \
"type": "gnocchi_resources_threshold", "resource_id": INSTANCE_ID1, \
"resource_type": "instance", "aggregation_method": "last"}, \
{"threshold": 500, "metric": "memory.usage", \
"type": "gnocchi_resources_threshold", "resource_id": INSTANCE_ID2, \
"resource_type": "instance", "aggregation_method": "last"}]}' \
```

This creates an alarm which will fire if either *INSTANCE\_ID1* or *INSTANCE\_ID2* uses more than 500MB memory.

### 11.1.8 References

Aodh Alarms: <https://docs.openstack.org/aodh/train/admin/telemetry-alarms.html>

Gnocchi Documentation: [https://gnocchi.xyz/stable\\_4.2/rest.html](https://gnocchi.xyz/stable_4.2/rest.html)

## 11.2 Gnocchi

**Warning:** This component is currently disabled and not available.

Gnocchi processes data from Ceilometer and stores the processed data which can be accessed by users. It correlates measurements from Ceilometer to **resources** and **metrics**.

**Metric:** A resource property which can be measured. For example the CPU, memory usage etc.

**Resource:** These can be instances, volumes, networks etc.

### 11.2.1 gnocchiclient

The gnocchi command line tool can be installed using pip:

```
pip install gnocchiclient
```

This provides the following Openstack commands:

```
openstack [metric-command]

metric aggregates
metric archive-policy create
metric archive-policy delete
metric archive-policy list
metric archive-policy show
metric archive-policy update
metric archive-policy-rule create
metric archive-policy-rule delete
metric archive-policy-rule list
metric archive-policy-rule show
metric benchmark measures add
metric benchmark measures show
metric benchmark metric create
metric benchmark metric show
metric capabilities list
metric create
metric delete
metric list
metric measures add
metric measures aggregation
metric measures batch-metrics
metric measures batch-resources-metrics
metric measures show
metric resource batch delete
```

(continues on next page)

(continued from previous page)

```
metric resource create
metric resource delete
metric resource history
metric resource list
metric resource search
metric resource show
metric resource update
metric resource-type create
metric resource-type delete
metric resource-type list
metric resource-type show
metric resource-type update
metric server version
metric show
metric status
```

## 11.2.2 Metrics

Metrics consist of:

- Name: unique name for the metric
- Unit (e.g. MB, ns, B/s)
- Resource ID - where the measurement is from
- Unique User ID (UUID): Identifies the metric
- Archive policy: stores and aggregates the measures

Archive policies determine how long aggregates are kept and how they are aggregated. The archive policies can be seen using the command:

```
openstack metric archive-policy list
```

```
+-----+-----+-----+-----+
↪-----+
| name          | back_window | definition                                     | ↪
↪aggregation_methods |
+-----+-----+-----+-----+
↪-----+
| ceilometer-high |             0 | - points: 3600, timespan: 1:00:00,          | mean ↪
↪          |             | | granularity: 0:00:01                        | ↪
↪          |             | | - points: 1440, timespan: 1 day,           | ↪
↪          |             | | 0:00:00, granularity: 0:01:00              | ↪
↪          |             | | - points: 8760, timespan: 365 days,        | ↪
↪          |             | | 0:00:00, granularity: 1:00:00              | ↪
↪          |             | | - points: 3600, timespan: 1:00:00,          | ↪
| ceilometer-high-rate |             0 | - points: 3600, timespan: 1:00:00,          | ↪
```

(continues on next page)

(continued from previous page)

↪rate:mean, mean			granularity: 0:00:01		↪
↪			- points: 1440, timespan: 1 day,		↪
↪			0:00:00, granularity: 0:01:00		↪
↪			- points: 8760, timespan: 365 days,		↪
↪			0:00:00, granularity: 1:00:00		↪
↪			- points: 8640, timespan: 30 days,	mean	↪
↪			0:00:00, granularity: 0:05:00		↪
↪			- points: 8640, timespan: 30 days,		↪
↪rate:mean, mean			0:00:00, granularity: 0:05:00		↪
↪					↪
+-----+-----+-----+-----+					
↪-----+					

This table shows each archive-policy and how the raw datapoints for each metric is stored. As an example, let's view the detail for one of the archive policies.

To view an archive policy, use the command:

```
openstack metric archive-policy show <archive-policy>
```

For example:

```
openstack archive-policy show ceilometer-high
```

↪-+		
Field	Value	↪
↪		
+-----+-----+		
↪-+		
aggregation_methods	mean	↪
↪		
back_window	0	↪
↪		
definition	- points: 3600, timespan: 1:00:00, granularity: 0:00:01	↪
↪		
	- points: 1440, timespan: 1 day, 0:00:00, granularity: 0:01:00	↪
↪		
	- points: 8760, timespan: 365 days, 0:00:00, granularity: 1:00:00	↪
↪1:00:00		
name	ceilometer-high	↪
↪		
+-----+-----+		
↪-+		

For the archive policy ceilometer-high:

- The **mean** is stored for each interval.
- Stores **one hour** of data in **one second** intervals. (3600 data points)
- Stores **one day** of data in **one minute** intervals. (1440 data points)
- Stores **one year** of data in **one hour** intervals. (8760 data points)

To view the list of metrics:

```
openstack metric list
```

This will show the metrics which have been created/visible to the user.

To view the metric resources:

```
openstack metric resource list
```

To view the properties of a specific metric:

```
openstack metric show <metric-id>
```

Metrics can be created by the user using the gnocchiclient plugin in the Openstack CLI:

```
openstack metric create [-h] [--resource-id RESOURCE_ID] [-f {json,shell,table,value,
↪yaml}]
                        [-c COLUMN] [--noindent] [--prefix PREFIX] [--max-width
↪<integer>]
                        [--fit-width] [--print-empty]
                        [--archive-policy-name ARCHIVE_POLICY_NAME] [--unit UNIT]
                        [METRIC_NAME]
```

**Note:** Once a metric has been created, the archive policy attribute for that metric is fixed and cannot be changed.

### 11.2.3 References:

Gnocchi Documentation: [https://gnocchi.xyz/stable\\_4.2/rest.html](https://gnocchi.xyz/stable_4.2/rest.html)

Telemetry Measurements for Train: <https://docs.openstack.org/ceilometer/train/admin/telemetry-measurements.html>

Gnocchi Aggregation: <https://medium.com/@berndbausch/how-i-learned-to-stop-worrying-and-love-gnocchi-aggregation-c98dfa2e20>

Gnocchi Glossary: [https://gnocchi.xyz/stable\\_4.2/glossary.html](https://gnocchi.xyz/stable_4.2/glossary.html)

## 11.3 Monitoring Virtual Machines: Gnocchi and OpenStack Aodh

**Warning:** This component is currently disabled and not available.

This document is a brief introduction to Aodh and Gnocchi.

We can monitor virtual machines by using alarms which fire when a threshold for a given measurement (e.g memory usage) has been met, when a VM has unexpectedly powered off, or when a VM has entered an error state.

In OpenStack, Aodh provides the ability to create, update and assign alarms to virtual machines.

To use alarms to monitor specific metrics from a virtual machine, Aodh uses metrics provided by Gnocchi.

### 11.3.1 Command Line Clients

The command line clients for Aodh and Gnocchi can be installed using pip:

```
pip install aodhclient
pip install gnocchiclient
```

### 11.3.2 Aodh

Aodh is the telemetry alarming service which triggers alarm when the collected data or metrics have met or broken defined rules.

Alarms are governed by defined rules and have three states that they can enter:

- **ok** - the rule has been evaluated as *false*
- **alarm** - the rule has been evaluated as *true*
- **insufficient\_data** - there is not enough data available to meaningfully determine the alarm state.

There are three types of alarms which can be created using Aodh:

- **Threshold Alarms:** these alarms move to an **alarm** state when a given threshold has been met or exceeded. An example of a threshold alarm would be a memory usage alarm, which uses the memory usage metric of the specific VM to determine the alarm's state. These alarms will move back to an **ok** state when the threshold is no longer being exceeded.
- **Event Alarms:** these alarms move to an **alarm** state when a given event has occurred to a resource e.g a VM has unexpectedly powered off. These alarms only move to the **alarm** state once and will need their status updated manually to 'reset' the alarm.

**Note:** Event alarms will always start in an **insufficient\_data** state once created and will only change state when the event has occurred.

- **Composite Alarms:** these alarms use a combination of rules to determine the state of an alarm. These rules are combined with logical operators.

Please see the **Create Aodh Alarms** documentation for more information on how to create Aodh Alarms in the command line.

### 11.3.3 Gnocchi

Gnocchi is a time series database which process data from the telemetry service Ceilometer. It correlates measurements from Ceilometer to **resources** and **metrics**.

**Metrics** are measurable quantities (e.g CPU time, memory usage etc) which are sampled from objects called **resources**. These resources could be instances (VMs), storage volume etc.

The gnocchi CLI enables us to use openstack commands to access and work with some of the metrics being stored.

```
openstack metric <command>

#The complete list of metric commands can be viewed using
openstack metric --help
```

## Metrics

A metric is a measurable resource property. This property could be memory usage, the amount of CPU time used, the number of vCPUs etc.

Metrics consist of:

- **Name:** a unique name for the metric
- **Unit:** e.g MB, ns, B/s
- **Resource ID:** the resource where the measurement is from
- **Unique User ID (UUID):** a unique identifier for the metric
- **Archive Policy:** The policy for aggregating and storing the measurement for the metric.

In OpenStack, Ceilometer automatically populates Gnocchi with metrics and resources.

The list of measurements from Ceilometer that Gnocchi can aggregate and store data from can be found here:

<https://docs.openstack.org/ceilometer/train/admin/telemetry-measurements.html>

**Note:** Since the Stein Release of OpenStack, the CPU utilization meter has been deprecated and the meter `cpu_util` cannot be used. However, gnocchi does provide a way to get the CPU utilization of a resource manually.

For example, metrics which are collected include:

Metric	Unit
network.incoming.bytes	B
network.incoming.packets	packet
network.outgoing.bytes	B
network.outgoing.packets	packet
disk.device.read.bytes	B
disk.device.read.requests	request
disk.device.write.bytes	B
disk.device.write.requests	request
cpu	ns
disk.ephemeral.size	GB
disk.root.size	GB
memory.usage	MB
memory	MB

(continues on next page)

(continued from previous page)

vcpus	vcpu	
+-----+	+-----+	+-----+

## CPU Utilization

Although the `cpu_util` meter has been deprecated since the OpenStack Stein release, we can use `gnocchi` to calculate the CPU utilization of a VM manually.

We can use the command `gnocchi aggregates <options>` to do this.

To calculate the CPU utilization of a VM, we can use the following command:

```
gnocchi aggregates '(* (/ (aggregate rate:mean (metric cpu mean)) 300000000000) 100)'
↪ id=INSTANCE_ID
```

This will return a table similar to the following:

name	timestamp	granularity	value
aggregated	2020-07-13T07:45:00+00:00	300.0	3.2666666666666666
aggregated	2020-07-13T07:50:00+00:00	300.0	1.1666666666666667
aggregated	2020-07-13T08:00:00+00:00	300.0	2.8666666666666667
aggregated	2020-07-13T08:15:00+00:00	300.0	25.866666666666667
aggregated	2020-07-13T08:25:00+00:00	300.0	2.2166666666666667
aggregated	2020-07-13T08:30:00+00:00	300.0	1.05
aggregated	2020-07-13T08:35:00+00:00	300.0	1.0333333333333332
aggregated	2020-07-13T08:40:00+00:00	300.0	1.05
aggregated	2020-07-13T08:45:00+00:00	300.0	1.95
aggregated	2020-07-13T08:50:00+00:00	300.0	1.1666666666666667
aggregated	2020-07-13T08:55:00+00:00	300.0	1.1333333333333333
aggregated	2020-07-13T09:00:00+00:00	300.0	1.7333333333333332
aggregated	2020-07-13T09:05:00+00:00	300.0	1.1166666666666667
aggregated	2020-07-13T09:10:00+00:00	300.0	1.2166666666666666
aggregated	2020-07-13T09:15:00+00:00	300.0	22.916666666666664
aggregated	2020-07-13T09:20:00+00:00	300.0	1.0999999999999999
aggregated	2020-07-13T09:25:00+00:00	300.0	0.9833333333333333
aggregated	2020-07-13T09:30:00+00:00	300.0	1.0666666666666667
aggregated	2020-07-13T09:35:00+00:00	300.0	1.0
aggregated	2020-07-13T09:40:00+00:00	300.0	1.0833333333333335
aggregated	2020-07-13T09:45:00+00:00	300.0	1.8833333333333333
aggregated	2020-07-13T09:50:00+00:00	300.0	1.25
aggregated	2020-07-13T09:55:00+00:00	300.0	1.0999999999999999
aggregated	2020-07-13T10:00:00+00:00	300.0	1.7833333333333332
aggregated	2020-07-13T10:05:00+00:00	300.0	1.1833333333333333
aggregated	2020-07-13T10:10:00+00:00	300.0	1.2333333333333334
aggregated	2020-07-13T10:15:00+00:00	300.0	22.55
aggregated	2020-07-13T10:20:00+00:00	300.0	1.0833333333333335
aggregated	2020-07-13T10:25:00+00:00	300.0	1.0666666666666667
aggregated	2020-07-13T10:30:00+00:00	300.0	1.1666666666666667
aggregated	2020-07-13T10:35:00+00:00	300.0	1.15

## Archive Policies

Archive policies are linked to every metric for each resource. These policies determines how many data points to collect over a given time period and the method for aggregating this data.

To view the list of archive policies, we can use the command:

```
openstack metric archive-policy list
```

This will return a table listing each archive policy and how the policies are defined. The definitions given to each policy determines how raw datapoints for metrics from OpenStack Ceilometer are collected and aggregated.

name	back_window	definition
aggregation_methods		
bool	3600	- points: 31536000, timespan: 365 days, 0:00:00, granularity: 0:00:01
ceilometer-high	0	- points: 3600, timespan: 1:00:00, granularity: 0:00:01
ceilometer-high-rate	0	- points: 3600, timespan: 1:00:00, granularity: 0:00:01
ceilometer-low	0	- points: 8640, timespan: 30 days, 0:00:00, granularity: 0:05:00
ceilometer-low-rate	0	- points: 8640, timespan: 30 days, 0:00:00, granularity: 0:05:00
high	0	- points: 3600, timespan: 1:00:00, granularity: 0:00:01
low	0	- points: 8640, timespan: 30 days, 0:00:00, granularity: 0:05:00
medium	0	- points: 10080, timespan: 7 days, 0:00:00, granularity: 0:01:00

We can view details for an archive policy using the command:

```
openstack metric archive-policy show <name>
```

This will return a table with information about the named archive policy. For example:

```
openstack archive-policy show ceilometer-high
```

This table shows each archive-policy and how the raw datapoints for each metric is stored. As an example, let's view the details for one of the archive policies.

```
openstack archive-policy show ceilometer-high
```

```
# Output for this archive policy
```

Field	Value
aggregation_methods	mean
back_window	0
definition	- points: 3600, timespan: 1:00:00, granularity: 0:00:01
	- points: 1440, timespan: 1 day, 0:00:00, granularity: 0:01:00
	- points: 8760, timespan: 365 days, 0:00:00, granularity: 1:00:00
name	ceilometer-high

In Gnocchi, **granularity** refers to the time interval between each aggregated data point. We can see from this table that for metrics collected using the archive policy ceilometer-high:

- The **mean** is stored for each interval.
- Stores **one hour** of data in **one second** intervals. (3600 data points)
- Stores **one day** of data in **one minute** intervals. (1440 data points)
- Stores **one year** of data in **one hour** intervals. (8760 data points)

**Note:** When creating threshold alarms which monitors metrics, it is important to check which *archive policy* they are using to collect the data. This is because each archive policy will have a different value for granularity. If a threshold alarm time interval is shorter than the granularity for that specific metric, the alarm will remain in an **insufficient\_data** state.

## Metric Commands

We can list the metrics in our project using the command:

```
openstack metric list

#Example Output
```

id	unit	resource_id	archive_policy/name	name
0009ecb6-ffdc-4b62-a870-10eee6be7c93	GB	1f2dbe24-1011-4039-8102-405d494eb14d	ceilometer-low	disk.root.size
00402d05-6af5-43c1-a1e9-03806ff04c3b	requests	request   136027f2-664f-5fb5-ba20-68800504f3f7	ceilometer-low-rate	disk.device.write.
00556936-8bd6-42f3-9e8f-17cb43824778	GB	8367a387-65e0-4071-b0ea-e5f434cc74ed	ceilometer-low	disk.root.size
006d7503-fb1a-4c66-91e1-0f2345b9cf92	GB	d1a8a32a-6904-4545-8ad4-7cde34101b61	ceilometer-low	disk.root.size
006e123e-18a7-4ce4-9a10-81b778399962	MB	f4ba2800-93c3-4bdf-b010-c9f5994e62aa	ceilometer-low	memory.usage
00815878-49ed-4e6a-9884-7556133306f1	packet	126e2936-7acd-5796-aeel-34946379a2de	ceilometer-low-rate	network.incoming.packets
008745cc-6c9b-41de-a09b-5b731c2d4ab1	MB	ce9638ca-c5b4-4824-b930-0d912a583f0b	ceilometer-low	memory
009fc9c6-9896-4388-9214-61c7ff332c53	vcpu	8808e24e-03bb-4e1b-aa28-d1c393f5e935	ceilometer-low	vcpus
00b46973-0df5-4a3c-b9d5-795a14cae0b8	GB	564973c9-5107-4851-a776-02156ff6f78a	ceilometer-low	disk.root.size
00ba0c03-89fe-4406-90b5-88fe94194c87	MB	dc5024b1-cd1a-4f25-99e2-a17471e15530	ceilometer-low	memory.usage
00c212c7-93a9-461b-a657-556181c9b4c1	time   sec	cc90947e-cad8-4890-a912-81459f718be0	ceilometer-low	compute.instance.booting.

# Note: This will return a list of every metric for every resource in the project

To view more information about the metric, we can use:

```
openstack metric show <uuid>

# Example output
```

Field	Value
archive_policy/name	ceilometer-low

(continues on next page)

(continued from previous page)

creator			
↪ e764d5abc65843fcb3bb060c80169871:4de86830e89b4a46b590536571b6ccd4			
id		f247f0ed-e5f0-4b72-95cc-b7771f984e83	
↪			
name		memory	
↪			
resource/created_by_project_id		4de86830e89b4a46b590536571b6ccd4	
↪			
resource/created_by_user_id		e764d5abc65843fcb3bb060c80169871	
↪			
resource/creator			
↪ e764d5abc65843fcb3bb060c80169871:4de86830e89b4a46b590536571b6ccd4			
resource/ended_at		None	
↪			
resource/id		69252292-8a40-400b-9446-8c1bfa9f471d	
↪			
resource/original_resource_id		69252292-8a40-400b-9446-8c1bfa9f471d	
↪			
resource/project_id		6a2f34e232744e59a5af8e105507f076	
↪			
resource/revision_end		None	
↪			
resource/revision_start		2020-10-13T14:01:40.979875+00:00	
↪			
resource/started_at		2020-08-06T13:48:43.421894+00:00	
↪			
resource/type		instance	
↪			
resource/user_id			
↪ 569d4d38222c86c68585e194b200eddea857137476dc76360b546b48f4319dde			
unit		MB	
↪			
+-----+			
↪ -----+			

We can view the metric resource list as well. The following command will return a table containing every single resource (instance, instance disk, etc) which metrics are attached to.

```
openstack metric resource show <resource-id>
# This will return every single resource in the project, this includes the volumes,
↪ disks etc associated to each VM
```

To view the measurements of a metric, we can use the command:

```
openstack metric measures show <metric-id>
```

To view the resource which has a metric linked to it, we can use the command:

```
openstack metric resource show <resource-id>

# This will also show the IDs for the metrics attached to that specific resource
+-----+
↪ -----+
```

(continues on next page)

(continued from previous page)

Field	Value
created_by_project_id	4de86830e89b4a46b590536571b6ccd4
created_by_user_id	e764d5abc65843fcb3bb060c80169871
creator	e764d5abc65843fcb3bb060c80169871:4de86830e89b4a46b590536571b6ccd4
ended_at	2020-09-28T13:00:17.994533+00:00
id	17ae2b28-7ed1-43e7-9099-e7e1134a10ad
metrics	compute.instance.booting.time: 8db0701c-1e3e-4af0-95b2-dc95cf1010c0
	cpu: f5d3c5ca-0f03-49b4-9ec0-e25d50cd7abd
	disk.ephemeral.size: 0b813829-7d48-4e96-a88c-472dc739b427
	disk.root.size: d8309af3-ef3d-4043-b80d-7f88b0b10d57
	memory.usage: 0a994a4c-7d9b-45d0-8e1f-82a9d6004b3e
	memory: 4e4b3247-2197-441f-a880-4af6edc89747
	vcpus: 65799f16-836d-445e-a395-d0bb6ac56ba5
original_resource_id	17ae2b28-7ed1-43e7-9099-e7e1134a10ad
project_id	PROJECT_ID
revision_end	None
revision_start	2020-09-28T13:00:31.836563+00:00
started_at	2020-09-28T12:52:51.968895+00:00
type	instance
user_id	USER_ID

To view the measurements of a metric, we can use the command:

```
openstack metric measures show <metric-id>
```

*#Example Output*

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

(continues on next page)

(continued from previous page)

timestamp	granularity	value
2020-09-13T16:00:00+01:00	300.0	8192.0
2020-09-13T17:00:00+01:00	300.0	8192.0
2020-09-13T18:00:00+01:00	300.0	8192.0
2020-09-13T19:00:00+01:00	300.0	8192.0
2020-09-13T20:00:00+01:00	300.0	8192.0
2020-09-13T21:00:00+01:00	300.0	8192.0
2020-09-13T22:00:00+01:00	300.0	8192.0
2020-09-13T23:00:00+01:00	300.0	8192.0
2020-09-14T00:00:00+01:00	300.0	8192.0
...	...	...

*#Note: This will return ALL measurements for the metric!*

To view the list of resource types:

```
openstack metric resource-type list
```

### 11.3.4 References

Gnocchi Documentation: [https://gnocchi.xyz/stable\\_4.2/rest.html](https://gnocchi.xyz/stable_4.2/rest.html)

Telemetry Measurements for Train: <https://docs.openstack.org/ceilometer/train/admin/telemetry-measurements.html>

Gnocchi Aggregation: <https://medium.com/@berndbausch/how-i-learned-to-stop-worrying-and-love-gnocchi-aggregation-c98dfa2e20>

Gnocchi Glossary: [https://gnocchi.xyz/stable\\_4.2/glossary.html](https://gnocchi.xyz/stable_4.2/glossary.html)

Aodh Alarms: <https://docs.openstack.org/aodh/train/admin/telemetry-alarms.html>

## KUBERNETES

Currently Cluster API is the preferred method for deploying Kubernetes on STFC cloud. We are developing an RKE2 + Terraform based deployment method which will be available in the near future.

### 12.1 ClusterAPI Introduction

Cluster API is a platform agnostic way of managing physical resources (i.e. VMs) using Kubernetes.

This provides a number of advantages including:

- Familiarity to experience K8s admins, as nodes act similarly to pods...etc.
- Tooling which directly supports Openstack, including resource creation and management.
- Support and fixes for newer Kubernetes versions without waiting for Openstack upgrades.

The Kubernetes image provided is based on [upstream's Ubuntu image](#), which has been forked to comply with UKRI security policy [here](#).

Unlike Magnum, the Kubernetes version supported is completely decoupled from Openstack, with upstream typically supporting the N-1 major version (where N is the latest release).

### 12.2 Setup a new cluster using Cluster API

**Warning:** Under Testing Cluster API is currently under release testing between interested users and the cloud team. It is not rated for production workloads at this time.

#### Contents

- *Setup a new cluster using Cluster API*
  - *Deployment Considerations*
    - \* *Management Machine*
    - \* *Account Security*
  - *Preparing to Deploy*
    - \* *Background*

- \* *Bootstrap Machine Prep*
- \* *Openstack Preparation*
- \* *clouds.yaml Prep*
- *Creating the cluster*
  - \* *Yaml Files*
  - \* *Configuring the cluster*
- *Moving the control plane*
  - \* *Moving from minikube cluster*
  - \* *minikube Shutdown*

## 12.2.1 Deployment Considerations

### Management Machine

For production workloads, it's recommended to create a dedicated cluster management machine. This should only be accessible to Cluster Administrators and holds a copy of the kubeconfig. The kubeconfig cannot be recovered (unlike Magnum) after generation. Additionally, a known working dedicated machine can be used to quickly gain access and perform recovery or upgrades as required.

### Account Security

A `clouds.yaml` file with the application credentials is also required. The credentials should not be unrestricted, as a compromised cluster would allow an attacker to create additional credentials. This file should be removed or restricted on shared machines to prevent unauthorized access.

## 12.2.2 Preparing to Deploy

### Background

A Kubernetes cluster is required to create the cluster. To break this “chicken-egg” problem a minikube cluster is created to bootstrap the main cluster. It is not recommended to use this cluster for any production workloads.

### Bootstrap Machine Prep

A Ubuntu machine is used to provide the minikube cluster. This should use the normal cloud Ubuntu image, not the stripped down CAPI image designed for nodes.

The following packages are required and can be installed and configured using the following commands:

- Docker is used to run a Kubernetes staging cluster locally

```
# Docker
sudo apt install -y docker.io
sudo usermod -aG docker $USER
```

- You will need to exit and login again if you have added yourself to the docker group (usermod). This is to pick up the new group membership

- Snap is used to install kubectl and Helm

```
sudo apt-get update && sudo apt-get install -y snapd
export PATH=$PATH:/snap/bin
sudo snap install kubectl --classic
sudo snap install helm --classic
```

- Install minikube and start a cluster:

```
wget https://storage.googleapis.com/minikube/releases/latest/minikube-linux-amd64
chmod +x minikube-linux-amd64
sudo mv minikube-linux-amd64 /usr/local/bin/minikube
minikube start --driver=docker
```

- Install clusterctl and the Openstack provider into your minikube cluster:

```
# YQ (Used by Cluster API)
sudo snap install yq
# Clusterctl
curl -L https://github.com/kubernetes-sigs/cluster-api/releases/download/v1.3.3/
  ↪clusterctl-linux-amd64 -o clusterctl
chmod +x ./clusterctl
sudo mv ./clusterctl /usr/local/bin/clusterctl
clusterctl init --infrastructure openstack
```

If you run into GitHub rate limiting you will have to generate a personal API token as described [here](#). This only requires the Repo scope, and is set on the CLI as follows

```
export GITHUB_TOKEN=<your token>
```

These setup steps only have to be completed once per management machine.

## Openstack Preparation

- Ensure a dedicated floating IP exists. If required, allocate an IP to the project from the External pool.
- Clone <https://github.com/stfc/cloud-capi-values>

## clouds.yaml Prep

- Generate your application credentials: *Openstack Application Credentials*. It is recommended you use Horizon (the web interface) to download the clouds.yaml file.

The clouds.yaml file should have the following format:

```
clouds:
  openstack:
    auth:
      auth_url: https://openstack.stfc.ac.uk:5000/v3
      application_credential_id: ""
      application_credential_secret: ""
    region_name: "RegionOne"
    interface: "public"
```

(continues on next page)

(continued from previous page)

```
identity_api_version: 3
auth_type: "v3applicationcredential"
```

- Add the UUID of the project you want to create the cluster in. This is the project ID under the Openstack section which is omitted by default. This can be found [here](#).

Your `clouds.yaml` should now look like:

```
clouds:
  openstack:
    auth:
      auth_url: https://openstack.stfc.ac.uk:5000/v3
      application_credential_id: ""
      application_credential_secret: ""
      project_id: ""
    region_name: "RegionOne"
    interface: "public"
    identity_api_version: 3
    auth_type: "v3applicationcredential"
```

- Place this file in the `cloud-capi-values` directory you cloned earlier.

### 12.2.3 Creating the cluster

#### Yaml Files

The configuration is spread across multiple yaml files to make it easier to manage. These are as follows:

- `values.yaml` contains the default values for the cluster using the STFC Cloud service. These should not be changed.
- `user-values.yaml` contains some values that must be set by the user. There are also optional values that can be changed too for advanced users.
- `flavors.yaml` contains the Openstack flavors to use for worker nodes. Common flavors are provided and can be uncommented and changed as required. By default the cluster will use `13.nano` workers by default if unspecified.
- `clouds.yaml` contains the Openstack application credentials. This file should be in the same directory as the other yaml files.

The cloud team will periodically update `flavors.yaml`, `values.yaml`, and `user-values.yaml` to reflect changes in the STFC Cloud service. These include new versions of Kubernetes or machine images, best practices, new flavors...etc. A user will pull these changes by running `git pull` in the `cloud-capi-values` directory in the future.

#### Configuring the cluster

- The mandatory values in `user-values.yaml` must be set. Optional values may also be changed as required.
- The `flavors.yaml` file contains the Openstack flavors to use for worker nodes. These can be changed as required but will use `13.nano` by default if unspecified.

```
cd cloud-capi-values
export CLUSTER_NAME="demo-cluster" # or your cluster name
```

(continues on next page)

(continued from previous page)

```
# Install the custom resource definitions (CRDs)
helm repo add capi-addons https://stackhpc.github.io/cluster-api-addon-provider
helm upgrade cluster-api-addon-provider capi-addons/cluster-api-addon-provider --install_
↪--version ">=0.1.0-dev.0.main.0,<0.1.0-dev.0.main.9999999999" --wait

# Deploy the cluster called "demo-cluster"
helm repo add capi https://stackhpc.github.io/capi-helm-charts
helm upgrade $CLUSTER_NAME capi/openstack-cluster --install -f values.yaml -f clouds.
↪yaml -f user-values.yaml -f flavors.yaml --wait
```

- Progress can be monitored with the following command in a separate terminal:

```
kubectl logs deploy/capo-controller-manager -n capo-system -f
```

- When the deployment is complete clusterctl will report the cluster as Ready: True

```
clusterctl describe cluster $CLUSTER_NAME
```

## 12.2.4 Moving the control plane

At this point the control plane is still on the minikube cluster. This is not recommended for long-lived or production workloads. We can pivot the cluster to self-manage:

**Warning:** After moving the control plane the kubeconfig cannot be retrieved if lost. Ensure a copy of the kubeconfig is placed into secure storage for production clusters.

### Moving from minikube cluster

- Install clusterctl into the new cluster and move the control plane

```
clusterctl get kubeconfig $CLUSTER_NAME > kubeconfig.$CLUSTER_NAME
clusterctl init --infrastructure openstack --kubeconfig=kubeconfig.$CLUSTER_NAME
clusterctl move --to-kubeconfig kubeconfig.$CLUSTER_NAME
```

- Ensure the control plane is now running on the new cluster:

```
kubectl get kubeadmcontrolplane --kubeconfig=kubeconfig.$CLUSTER_NAME
```

### minikube Shutdown

- Replace the existing minikube kubeconfig with the new cluster's kubeconfig

```
cp -v kubeconfig.$CLUSTER_NAME ~/.kube/config
# Ensure kubectl now uses the new kubeconfig displayed the correct nodes:
kubectl get nodes

# Update the cluster to ensure everything lines up with your helm chart
helm upgrade cluster-api-addon-provider capi-addons/cluster-api-addon-provider --install_
```

(continues on next page)

(continued from previous page)

```
↪--version ">=0.1.0-dev.0.main.0,<0.1.0-dev.0.main.999999999" --wait
helm upgrade $CLUSTER_NAME capi/openstack-cluster --install -f values.yaml -f clouds.
↪yaml -f user-values.yaml -f flavors.yaml --wait
```

```
# Check the cluster status
clusterctl describe cluster $CLUSTER_NAME
```

- Remove minikube bootstrap cluster

```
minikube delete
```

Your cluster is now complete

## 12.3 Openstack Load Balancers in External K8s configurations

**Warning:** The deployment setup for Cluster API and helm charts will automatically deploy Openstack load balancers. Attempting to deploy this method too will result in conflicts.

These instructions are for users who have deployed RKE2 / K3s or other K8s deployments without the helm chart.

### 12.3.1 Background

Load balancers are managed by the external Openstack controller driver. This is deployment agnostic, i.e. can be used on Cluster API, RKE and other K8s deployments.

### 12.3.2 Setup

- Follow the steps to setup the clouds file: *Setting Up Clouds.yaml*
- Clone the ClusterAPI Openstack scripts:

```
git clone https://github.com/kubernetes-sigs/cluster-api-provider-openstack
cd cluster-api-provider-openstack
```

- Generate the secret containing the clouds conf:

```
# Substitute <cloud> with the name from the clouds.yaml file
templates/create_cloud_conf.sh ~/.config/openstack/clouds.yaml <cloud> > /tmp/cloud.conf
kubectl create secret -n kube-system generic cloud-config --from-file=/tmp/cloud.conf
rm /tmp/cloud.conf
```

- Deploy the out of tree controller module

```
kubectl apply -f https://raw.githubusercontent.com/kubernetes/cloud-provider-openstack/
↪master/manifests/controller-manager/cloud-controller-manager-roles.yaml
kubectl apply -f https://raw.githubusercontent.com/kubernetes/cloud-provider-openstack/
↪master/manifests/controller-manager/cloud-controller-manager-role-bindings.yaml
kubectl apply -f https://raw.githubusercontent.com/kubernetes/cloud-provider-openstack/
↪master/manifests/controller-manager/openstack-cloud-controller-manager-ds.yaml
```

- Check the *Openstack controller manager* deploys:

```
kubect1 get pods -n kube-system
```

### 12.3.3 Usage

The Openstack Controller Manager will automatically register itself as a loadbalancer provider.

Deployments / charts which use the *loadbalancer* port type will automatically provision a load balancer within Openstack. Additionally, a floating IP can be specified (this must belong to the Openstack project) and will be used for an idempotent deployment.



## IMPROVING THIS DOCUMENTATION

You can help us to improve our documentation in several ways:

You can find the repository for this documentation at <https://github.com/stfc/cloud-docs>

If you have anything you would like to see documented then you can raise issues against the documentation and we will do our best to fill in the documentation.

Alternatively, you can email cloud support and we can open an issue on your behalf.

Any pull requests to improve or change the documentation, however minor, is welcome and greatly appreciated.

### 13.1 Setting Up a local development environment

The following assumes you are using Ubuntu 18.04 or 20.04 LTS through Windows Subsystem for Linux or natively with GitHub Desktop to manage interaction with GitHub.

Create a fork on GitHub and clone it into GitHub desktop.

First install the software necessary to build the repository

```
sudo apt-get update
sudo apt-get install build-essential python3-sphinx python3-sphinx-rtd-theme
```

Change into the directory you cloned the repository to.

Run the following to build a local copy of the documentation:

```
make html
```

Assuming that this succeeds, you can then open `./build/html/index.html` in your web browser

Make any changes you wish to and then build again and test.

A good resource for editing rst is here: <https://github.com/ralsina/rst-cheatsheet/blob/master/rst-cheatsheet.rst>

Once you are happy with your changes, commit the changes and publish and then create a pull request for us to review.

If you're struggling feel free to open a PR with your changes so far and we'll be glad to provide assistance



## 14.1 Introducing OpenStack Heat

OpenStack Heat is the Orchestration service for OpenStack and orchestrates infrastructure resources (such as VMs, floating IPs, volumes, networks etc.) for cloud applications. This is done using templates in the form of YAML files which contain the properties and the relationships between resources.

Heat Templates allow relationships to be defined between different resources, allowing Heat to call different OpenStack APIs to create different resources such as LBaaS Pools (Octavia), servers and server groups (Nova), alarms (Aodh), volumes (Cinder), networks and security groups (Neutron). Heat manages the lifecycle of the stack and when a stack needs to be updated, an updated template can be applied to the existing stack. These templates can be integrated with software configuration management tools such as Ansible and Puppet.

### 14.1.1 Architecture

Heat provides an AWS CloudFormation implementation for OpenStack. Heat provides integration of other core OpenStack components into a one-file template system. This system not only allows resources from most OpenStack Projects to be created in a single template, it also provides more functionality including auto scaling of VMs and nested stacks.

**heat:** A CLI that communicates with the heat-api to execute AWS CloudFormation APIs. End developer could use heat REST API directly.

**heat-api:** provides an OpenStack-native REST API that processes API requests by sending them to the heat engine over RPC

**heat-api-cfn:** provides AWS Query API that is compatible with AWS CloudFormation and processes API requests by sending them to the heat-engine over RPC

**heat-engine:** orchestrates the launching of templates and provides events back to API consumer.

### 14.1.2 Heat Commands

To use Heat commands in the command line, we need to install the following package:

```
pip install python-heatclient
```

To test that this has installed correctly, and that we are able to access the Heat API, we can run the command:

```
openstack stack list
```

*#This should return an empty line if there are no stacks in the project or a table.*

(continues on next page)

(continued from previous page)

↪ *similar to the following:*

↪ -----+-----+-----+↪			
↪ -----+-----+-----+↪			
ID	Stack Name	Stack Status	↪
↪ Creation Time	Updated Time		
↪ -----+-----+-----+↪			
a00fa2cd-3e29-489f-8d9f-f805956	software-deployment-test	CREATE_COMPLETE	↪
↪ 2020-12-09T08:34:15Z	None		
7d045			↪
↪			
1cb66fbd-1336-414b-b112-b0fcffe	spark-standalone-cluster	CREATE_COMPLETE	↪
↪ 2020-12-07T10:31:58Z	None		
b0645			↪
↪			
b7263b67-65c4-4333-abd0-7033afa	spark-stack-2	CREATE_FAILED	↪
↪ 2020-12-04T11:42:25Z	None		
961b6			↪
↪			
c9c10097-c275-4c44-9324-0eb2f7a	docker-script-test	CREATE_COMPLETE	↪
↪ 2020-12-02T16:39:54Z	None		
d60cf			↪
↪			
↪ -----+-----+-----+↪			
↪ -----+-----+-----+↪			

The following list is the list of commands from Heat which can be used in OpenStack:

*# Commands provided by Heat are of the form:*

openstack stack <command> <options>

```

stack abandon # abandon a stack and output results
stack adopt # adopt a stack
stack cancel # cancel create or update task for a stack
stack check # check stack and its resources
stack create # create a stack
stack delete # delete a stack
stack environment show #
stack event list # List stack events
stack event show # View stack event
stack export # export stack data json
stack failures list # List failed resources in a stack
stack file list # show a stack's files map
stack hook clear # clear resource hooks on a given stack
stack hook poll # list resources with pending hook for a stack
stack list # list stacks in the project
stack output list # list stack outputs
stack output show # view stack output
stack resource list # list stack resources
stack resource mark unhealthy # mark one of the stack resources as unhealthy
stack resource metadata # view metadata for stack's resource

```

(continues on next page)

(continued from previous page)

```

stack resource show # view details about a stack's resource
stack resource signal # signal a resource with optional data (JSON data)
stack resume # resume a stack
stack show # view details about a stack
stack snapshot create # create a snapshot of the stack
stack snapshot delete # delete stack snapshot
stack snapshot list # list stack snapshots
stack snapshot restore # restore stack snapshot
stack snapshot show # view details of a stack snapshot
stack suspend # suspend a stack
stack template show # view stack template
stack update # update a stack using an updated template

```

### 14.1.3 Stacks

**Stacks:** a collection of resources and their associated configuration

**Template:** A YAML file defining the resources which make up the stack. In OpenStack, templates follow the Heat Orchestration Template (HOT) format.

**Note:** While Heat can interpret CFN (CloudFormation) Templates, they are *\_not\_* backwards compatible with Heat Orchestrated Templates. It is recommended to use Heat Orchestrated Templates to create stacks.

#### Heat Orchestrated Templates

Heat Orchestrated Templates are YAML files that instruct Heat which resources to create and the relationships between resources. Ansible has documentation on how to write YAML files that can be found here: [https://docs.ansible.com/ansible/latest/reference\\_appendices/YAMLSyntax.html](https://docs.ansible.com/ansible/latest/reference_appendices/YAMLSyntax.html)

Heat Templates consist of seven sections:

1. **heat\_template\_version:** (**required**) Indicates which format and features are used and supported when creating the stack.
2. **description:** (optional) Description of the stack template. It is recommended to include a description in templates to describe what users can do with the template.
3. **parameter\_groups:** (optional) Defines how to group input parameters and the order of the parameters.
4. **parameters:** (optional) Defines input parameters. This section can be omitted if there are no input values required.
5. **resources:** (**required**) Defines resources in the template. At least one resource should be defined in a HOT template.
6. **outputs:** (optional) Defines output parameters available to users once the template has been instantiated.
7. **conditions:** (optional) Includes statements which can be used to apply conditions to a resource, for example a resource is created only when a property is defined or when another resource has been created first.

The structure of a HOT template is given as:

```

heat_template_version: 2018-08-31 #OpenStack Version we want to use.
                                   #Here, we want to use template for the Rocky release.
↪onwards

```

(continues on next page)

(continued from previous page)

```

description: #description of the template

parameter_groups: #declares the parameter group and order.
    #This is not a compulsory section, however it is useful for grouping
    #parameters together when building more complex templates.

parameters: #declares the parameters for resources

resources: #declares the template resources
# e.g. alarms, floating IPs, instances etc.

outputs: #declares the output of the stack

conditions: #declares any conditions on the stack

```

Please see the documentation **Create a Heat Stack** (<https://stfc-cloud-docs.readthedocs.io/en/latest/howto/CreateAHeatStack.html>) for an introduction to creating a stack using a HOT template.

## Rocky Heat Templates

Templates which use:

```

heat_template_version: 2018-08-31

#Or

heat_template_version: rocky

```

Indicate that the template is a HOT template and has features added and/or removed up to the Queens Release. The list of supported functions in a Rocky Heat Template is:

```

digest      # allows for performing digest operations on a given value
filter      # removes values from list
get_attr    # references an attribute of a resource
get_file    # returns the content of a file into the template. Use to include files,
↳containing scripts or configuration files
get_param   # references an input parameter of a template
get_resource # references another resource in the same template
list_join   # joins a string with the given delimiter
make_url    # builds URLs
list_concat # concatenates lists together
list_concat_unique # behaves identically to list_concat. Only removes repeating items,
↳of lists
contains    # checks whether a specific value is in a sequence
map_merge   # merges maps together
map_replace # performs key/value replacements on existing mapping
repeat      # allows for dynamically transforming lists by iterating over the contents of,
↳one or more source lists and replacing list elements in the template
resource_facade # retrieves data in a parent provider template. A facade is a custom,
↳definition of a resource from a provider template

```

(continues on next page)

(continued from previous page)

```

str_replace    # constructs strings by providing a template string with placeholders and
↳ a list of mappings to assign values to those placeholders at runtime
str_replace_strict  # similar to str_replace, only an error is raised if any params
↳ are not present in template
str_replace_vstrict  # similar to str_replace, only an error is raised if any params
↳ are empty
str_split    # allows for a string to be split into a list by providing an arbitrary
↳ delimiter
yaql    # evaluates yaql expression on given data
if      # returns corresponding value based on evaluation of a condition

```

For more details about these intrinsic functions, please see the following documentation: [https://docs.openstack.org/heat/train/template\\_guide/hot\\_spec.html#get-attr](https://docs.openstack.org/heat/train/template_guide/hot_spec.html#get-attr)

The list of supported condition functions is:

```

equals        # compares whether two values are equal
get_param     # references input parameter of a template
not           # acts as a NOT operator
and           # acts as an AND operator
or            # acts as an OR operator
yaql          # evaluates yaql expression on given data
contains      # checks whether a specific value is in a sequence

```

## Pseudo Parameters

As well as parameters defined by the template author. Heat creates three parameters for every stack:

```

OS::stack_name # stack name
OS::stack_id   # stack identifier
OS::project_id # project identifier

```

These parameters are accessible using `get_param` function.

## 14.1.4 References

<https://www.cisco.com/c/dam/en/us/products/collateral/cloud-systems-management/metacloud/newbie-tutorial-heat.pdf>

<https://docs.aws.amazon.com/AWSCloudFormation/latest/APIReference/Welcome.html?r=7078>

[https://docs.openstack.org/heat/train/developing\\_guides/architecture.html](https://docs.openstack.org/heat/train/developing_guides/architecture.html)

[https://docs.openstack.org/heat/train/template\\_guide/openstack.html](https://docs.openstack.org/heat/train/template_guide/openstack.html)

<https://docs.openstack.org/heat/latest/doc-heat.pdf>

[https://docs.openstack.org/heat/train/template\\_guide/hot\\_spec.html#hot-spec](https://docs.openstack.org/heat/train/template_guide/hot_spec.html#hot-spec)

[https://docs.ansible.com/ansible/latest/reference\\_appendices/YAMLSyntax.html](https://docs.ansible.com/ansible/latest/reference_appendices/YAMLSyntax.html)

## 14.2 Assign Floating IPs to VMs in a Heat Stack

Floating IPs can be assigned to VMs that are on private networks in order for them to be accessible from an external network.

This document assumes that Floating IPs have been assigned to your project. If you do not have any floating IPs in your project, please contact the cloud team at [cloud-support@stfc.ac.uk](mailto:cloud-support@stfc.ac.uk)

**There are two similar methods for assigning a floating IP to a virtual machine in a heat stack.**

1. Use the resource `OS::Neutron::FloatingIPAssociation`.
2. Add the ID of the floating IP in the networks property of the `OS::Nova::Server` Resource

For both methods, `OS::Neutron::Port` is used in order to define a port on the VM on which to attach the floating IP.

To get the ID of the floating IPs, use the command:

```
openstack floating ip list
```

This will return a list of floating IPs containing the IP address, pool, port, and ID.

### 14.2.1 Using `OS::Neutron::FloatingIPAssociation`

```
heat_template_version: <template-version>

parameters:
  <define parameters here>

resources:
  private_network_port: # define the VM port which will be used to attach the floating IP
    type: OS::Neutron::Port
    properties:
      network_id: <private network ID> #private network ID
      security_groups: [{get_param: security_group_id}]

  test_VM:
    type: OS::Nova::Server
    properties:
      image: {get_param: image_id}
      flavor: {get_param: flavor_id}
      key_name: {get_param: key_name}
      networks:
        - port: {get_resource: private_network_port}

  floating_ip_association:
    type: OS::Neutron::FloatingIPAssociation
    properties:
      floatingip_id: <ID of Floating IP address> #ID of IP address to assign to VM
      port_id: {get_resource: private_network_port} #port to attach floating IP
```

### 14.2.2 Using the Network Property in OS::Nova::Server

```
heat_template_version: <template-version>

parameters:
  <define parameters>

resources:
  private_network_port:
    type: OS::Neutron::Port
    properties:
      network_id: <private-network-id> #private network ID
      security_groups: [{get_param: security_group_id}]

  test_VM:
    type: OS::Nova::Server
    properties:
      image: {get_param: image_id}
      flavor: {get_param: flavor_id}
      key_name: {get_param: key_name}
      networks: [{network: <private-network-id>, port: {get_resource: private_network_
↪port}}, floating_ip: <floating-ip-id> ]]
```

## 14.3 AutoScaling in a Heat Stack

Sometimes we may need to have a stack that can respond when a group of servers are using a lot or little resources, such as memory usage. For example if a group of servers exceed a given memory usage threshold, we want that group of resources to scale up. This documentation will go through autoscaling, and how autoscaling could be implemented in a Heat stack.

To create an autoscaling stack we need:

- **AutoScaling Group:** A group of servers defined so that the number of servers in the group and be increased or decreased.
- **Alarms:** Alarms created using OpenStack Aodh to monitor the resource usage of the VMs in the autoscaling group. For example, we could create an alarm to monitor memory usage and alarm if the autoscaling group exceeds the alarm's threshold.
- **Scaling Policies:** Policies which are executed when an Aodh Alarm is triggered. When an alarm is triggered, the scaling policy attached to that alarm will instruct the autoscaling group to change in size, either increasing or decreasing the number of VMs.

### 14.3.1 Heat Resources

This section will cover the resources available in Heat that are required for creating an autoscaling stack.

AutoScaling involves resources from:

- **Heat:** For creating an autoscaling group and defining the scaling policies
- **Aodh:** For alarm creation
- **Gnocchi:** For metrics that are used in threshold alarms

#### OS::Heat::AutoScalingGroup

This is an autoscaling group which can scale resources. This group can create the desired number of similar resources and we can define the minimum and maximum count for the given resource.

```
the_resource:
  type: OS::Heat::AutoScalingGroup
  properties:
    #required
    max_size: Integer # maximum number of resources in the group
    min_size: Integer # minimum number of resources in the group
    resource: {...} # resource definition for the resources in the group, written in HOT_
    ↪ (Heat Orchestrated Template) format
    #optional
    desired_capacity: Integer # desired initial number of resources
    cooldown: Integer # cool down period in seconds
    rolling_updates: {"min_in_service": Integer, "max_batch_size": Integer, "pause_time
    ↪": Number} # policy for rolling updates in the group, defaults to: {"min_in_service":
    ↪ 0, "max_batch_size": 1, "pause_time": 0}
    # min_in_service: minimum number of resources in service while rolling updates are_
    ↪ executed
    # max_batch_size: maximum number of resources to replace at once
    # pause_time: number of seconds to wait between batches of updates
```

For example:

```
autoscaling-group:
  type: OS::Heat::AutoScalingGroup
  properties:
    min_size: 1
    max_size: 3
    resource:
      type: server.yaml #Refers to a Heat Template for creating a VM
      properties:
        flavor: {get_param: flavor}
        image: {get_param: image}
        key_name: {get_param: key_name}
        network: {get_param: network}
        metadata: {"metering.server_group": {get_param: "OS::stack_id"}}
```

## OS::Heat::ScalingPolicy

```

the_resource:
  type: OS::Heat::ScalingPolicy
  properties:
    # required
    adjustment_type: String # Type of adjustment. Allowed values: "change_in_capacity",
↪ "exact_capacity", "percent_change_in_capacity"
    auto_scaling_group_id: String # AutoScaling Group ID to apply policy to
    scaling_adjustment: Number # Size of adjustment
    # Optional
    cooldown: Number # cooldown period, in seconds
    min_adjustment_step: Integer # minimum number of resources that are added or removed
↪ when the AutoScalingGroup scales up or down. Only used if specifying percent_change_in_
↪ capacity for adjustment_type property

```

For example:

```

scaleup_policy:
  type: OS::Heat::ScalingPolicy
  properties:
    adjustment_type: change_in_capacity
    auto_scaling_group_id: {get_resource: autoscaling-group}
    cooldown: 60
    scaling_adjustment: 1

scaledown_policy:
  type: OS::Heat::ScalingPolicy
  properties:
    adjustment_type: change_in_capacity
    auto_scaling_group_id: {get_resource: autoscaling-group}
    cooldown: 60
    scaling_adjustment: -1

```

## OS::Aodh::GnocchiAggregationByResourcesAlarm

This resource creates an alarm as an aggregation of resources alarm. This alarm is a threshold alarm monitoring the aggregated metrics of the members of the autoscaling group defined above. Gnocchi provides the metrics which Aodh uses to determine whether an alarm should be triggered.

```

the_resource:
  type: OS::Aodh::GnocchiAggregationByResourcesAlarm
  properties:
    # required
    metric: String # metric name watched by the alarm
    query: String # query to filter the metrics
    resource_type: String # resource type
    threshold: Number # threshold to evaluate against

    # optional
    aggregation_method: String # method to compare to the threshold
    alarm_actions: [Value, Value, ...] # list of webhooks to invoke when state

```

(continues on next page)

(continued from previous page)

```

↳ transitions to alarm
    alarm_queues: [String, String, ...] # list of Zaqr queues to post to when state_
↳ transitions to alarm
    comparison_operator: String # operator used to compare specified statistic with_
↳ threshold. Allowed values: "le", "ge", "eq", "lt", "gt", "ne"
    description: String # alarm description
    enabled: Boolean # Defaults to true. Determines if alarm evaluation is enabled
    evaluation_periods: Integer # number of periods to evaluate over
    granularity: Integer # time range in seconds
    insufficient_data_actions: [Value, Value, ...] # list of webhooks to invoke when_
↳ state transitions to insufficient data
    insufficient_data_queues: [String, String, ...] # list of Zaqr queues to post to_
↳ when state transitions to alarm
    ok_actions: [Value, Value, ...] # list of webhooks to invoke when state transitions_
↳ to ok
    ok_queues: [String, String, ...] # list of Zaqr queues to post to when state_
↳ transitions to ok
    repeat_actions: Boolean # Defaults to True. False to trigger actions when the_
↳ threshold is reached AND the alarm has changed state
    severity: String # severity of alarm. Allowed values: "low", "moderate", "critical"
    time_constraints: [{"name": String, "start": String, "description": String, "duration
↳ ": Integer, "timezone": String}, {"name": String, "start": String, "description":_
↳ String, "duration": Integer, "timezone": String}, ...] # Describe time constraints for_
↳ alarm, defaults to []
    # description: description for time constraints
    # duration: duration for time constraint
    # name: name for time constraint
    # start: start time for time constraint. A CRON expression property
    # timezone: Timezone for the time constraint.

```

For example, for our autoscaling stack we could define the alarms in the following way:

```

memory_alarm_high:
  type: OS::Aodh::GnocchiAggregationByResourcesAlarm
  properties:
    description: Scale up if memory > 1000 MB
    metric: memory.usage
    aggregation_method: mean
    granularity: 300
    evaluation_periods: 1
    threshold: 1000
    resource_type: instance
    comparison_operator: gt
    query:
      list_join:
        - ''
        - - {'=': {server_group: {get_param: "OS::stack_id"}}}
    alarm_actions:
      - get_attr: [scaleup_policy, signal_url]

memory_alarm_low:
  type: OS::Aodh::GnocchiAggregationByResourcesAlarm

```

(continues on next page)

(continued from previous page)

```

properties:
  description: Scale down if memory < 200MB
  metric: memory.usage
  aggregation_method: mean
  granularity: 300
  evaluation_periods: 1
  threshold: 200
  resource_type: instance
  comparison_operator: lt
  query:
    list_join:
      - ''
      - - {'=': {server_group : {get_param: "OS::stack_id"}}}
  alarm_actions:
    - get_attr: [scaledown_policy, signal_url]

```

### 14.3.2 References

[https://docs.openstack.org/heat/train/template\\_guide/openstack.html](https://docs.openstack.org/heat/train/template_guide/openstack.html)

<https://ibm-blue-box-help.github.io/help-documentation/heat/autoscaling-with-heat/>

<https://github.com/openstack/heat-templates/blob/master/hot/autoscaling.yaml>

<https://bhujaykbhatta.wordpress.com/2018/01/18/auto-scaling-in-openstack-using-heat-gnocchi-and-aodh/>

## 14.4 Creating a LAMP Stack

LAMP stacks contain a group of software consisting of:

- Linux OS
- Apache web server
- mySQL Database
- PHP

### 14.4.1 Templates

The following templates shows how an instance can be configured using a bash script to install Apache, mySQL, and PHP. The bash script is executed during the VM's cloud-init phase.

**Note:** During the cloud-init phase, root executes any user-scripts.

> During the execution of the scripts, we will be 'holding' some of the packages to stop them being upgraded during the *apt-get upgrade* step. This is because the upgrade requires a user response for the packages. These can be upgraded by the user after VM configuration is complete by running the upgrade command.

## Using OS::Heat::SoftwareConfig

The heat resource OS::Heat::SoftwareConfig has been used to define the user script.

OS::Heat::SoftwareConfig

```
heat_template_version: 2018-03-02

parameters:
  key_name:
    type: string
    default: <key-pair-name>
    description: Key Pair to use in order to SSH into the instance.
  image_id:
    type: string
    default: <image-id>
    description: The image for the instance
  flavor_id:
    type: string
    default: <flavor-id>
    description: The flavor for the instance
  security_group_id:
    type: string
    default: <security-group-id>

resources:
  test_script:
    type: OS::Heat::SoftwareConfig
    properties:
      group: ungrouped
      config: |
        #!/bin/bash -v
        apt-get update
        apt-mark hold libpam-krb5 libpam-modules libpam-modules-bin libpam-runtime_
↪libpam-systemd libpam0g
        apt-get -y upgrade
        apt-mark unhold libpam-krb5 libpam-modules libpam-modules-bin libpam-runtime_
↪libpam-systemd libpam0g
        # mysql-server is installed at this stage (can install after VM creation)
        apt-get install -y apache2 php libapache2-mod-php php-mysql php-gd mysql-server

  test_VM:
    type: OS::Nova::Server
    properties:
      image: {get_param: image_id}
      flavor: {get_param: flavor_id}
      key_name: {get_param: key_name}
      networks:
        - network: Internal
      security_groups:
        - {get_param: security_group_id}
      user_data_format: SOFTWARE_CONFIG
      user_data: {get_resource: test_script}
```

## Using OS::Nova::Server

The bash script can be placed alternatively in the `user_data` property of the `OS::Nova::Server` resource.

```
heat_template_version: <template-version>

parameters:
  <define parameters>

resources:
  VM:
    type: OS::Nova::Server
    properties:
      image: {get_param: image_id}
      flavor: {get_param: flavor_id}
      key_name: {get_param: key_name}
      networks:
        - network: <network-name>
      security_groups:
        - {get_param: security_group_id}
      user_data: |
        #!/bin/bash -v
        apt-get update
        apt-mark hold libpam-krb5 libpam-modules libpam-modules-bin libpam-runtime_
↪libpam-systemd libpam0g
        apt-get -y upgrade
        apt-mark unhold libpam-krb5 libpam-modules libpam-modules-bin libpam-runtime_
↪libpam-systemd libpam0g
        # mysql-server is installed at this stage (can install after VM creation)
        apt-get install -y apache2 php libapache2-mod-php php-mysql php-gd mysql-server
```

When a Bash Script becomes too long or complex, the `get_file` function can be used to retrieve and execute the bash script:

```
heat_template_version: <template-version>

parameters:
  <define parameters>

resources:
  VM:
    type: OS::Nova::Server
    properties:
      image: {get_param: image_id}
      flavor: {get_param: flavor_id}
      key_name: {get_param: key_name}
      networks:
        - network: <network-name>
      security_groups:
        - {get_param: security_group_id}
      user_data_format: RAW
      user_data: {get_file: bash-script.sh}
```

> Note: `user_data_format` is required for defining how the `user_data` should be formatted for the server.

without this parameter when defining a bash script this way, cloud init returns errors in the log and cannot run the script.

The function `str_replace` can be used in order to set variable values in the bash script based on parameters or resources in the stack.

Example from Openstack: [https://docs.openstack.org/heat/rocky/template\\_guide/software\\_deployment.html](https://docs.openstack.org/heat/rocky/template_guide/software_deployment.html)

```
heat_template_version: <template-version>

parameters:
  <define parameters>

resources:
  VM:
    type: OS::Nova::Server
    properties:
      image: {get_param: image_id}
      flavor: {get_param: flavor_id}
      key_name: {get_param: key_name}
      networks:
        - network: <network-name>
      security_groups:
        - {get_param: security_group_id}
      user_data:
        str_replace:
          template: |
            #!/bin/bash
            # ...
        params:
          $F00: {get_param: foo}
```

> **Note:** If a stack-update is performed and any changes have been made to the stack update, then the server will be deleted and replaced.

> If you are using the above bash scripts to install mysql-server as well as the other components of the stack, it is best to also run `mysql_secure_installation` as well. To automate `mysql_secure_installation` steps, please see the page **Installing and setting up MySQL database in a Stack**

## References:

[https://docs.openstack.org/heat/rocky/template\\_guide/software\\_deployment.html](https://docs.openstack.org/heat/rocky/template_guide/software_deployment.html)

<https://www.digitalocean.com/community/tutorials/how-to-install-linux-apache-mysql-php-lamp-stack-on-ubuntu-16-04>

## 14.5 Create LEMP Stack

LEMP Stack consists of:

- Linux
- Nginx
- MySQL
- PHP

### 14.5.1 Templates

Similar to installing a LAMP stack, you create a template that will execute a bash script to install LEMP on a server. The bash script below can be used in conjunction with the templates on the **Create a LAMP stack** docs at <https://stfc-cloud-docs.readthedocs.io/en/latest/Heat/CreateALAMPStack.html>.

### 14.5.2 Bash Script

A bash script can be added to a stack template so that when a virtual machine is built, it will automatically set up the VM with the components for a LEMP stack. This script also shows how to automate mysql\_secure\_installation as well.

```
#!/bin/bash -v
apt-get update
# packages for PAM should not be updated by using a bash script
# these return prompts to ask if the configuration on the VM can be changed
# and the script becomes 'stuck'
#To overcome this, use apt-mark hold <packages>
apt-mark hold libpam-krb5 libpam-modules libpam-modules-bin libpam-runtime libpam-
↳systemd libpam0g
apt-get -y upgrade
#Then unhold the packages
apt-mark unhold libpam-krb5 libpam-modules libpam-modules-bin libpam-runtime libpam-
↳systemd libpam0g

#Install nginx
apt-get install -y nginx

# In this example, we will install and set up mysql. Alternatively this step can be done
↳after VM creation
apt-get install -y expect #to run interactive script inside bash shell

#Install MySQL and set up root access
apt-get install -y mysql-server
SECURE_MYSQL=$(expect -c "
set timeout 5
spawn mysql_secure_installation

expect \"Press y|Y for Yes, any other key for No:\"
send \"n\r\"

expect \"Please set the password for root here.\"
```

(continues on next page)

(continued from previous page)

```

send \"temporarypw\r\"

expect \"Re-enter password:\"
send \"temporarypw\r\"

expect \"Remove anonymous users? (Press y|Y for Yes, any other key for No):\"
send \"y\r\"

expect \"Disallow root login remotely? (Press y|Y for Yes, any other key for No):\"
send \"y\r\"

expect \"Remove test database and access to it? (Press y|Y for Yes, any other key for ↵
↵No):\"
send \"y\r\"

expect \"Reload privilege tables now? (Press y|Y for Yes, any other key for No)\"
send \"y\r\"
expect eof
")
echo "$SECURE_MYSQL"

mysql <<EOF

SELECT user,authentication_string,plugin,host FROM mysql.user;
ALTER USER 'root'@'localhost' IDENTIFIED WITH mysql_native_password BY 'secret';
FLUSH PRIVILEGES;
SELECT user,authentication_string,plugin,host FROM mysql.user;

EOF

#add-apt-repository universe #already available for ubuntu bionic
#install PHP and PHP packages
apt-get install -y php-fpm php-mysql

echo "All components for LEMP stack have been installed!"

```

This script sets up the root password as a temporary password. If you set up a temporary password, this **must** be changed once the root user has signed into mySQL. Alternatively, the Heat resource `OS::Heat::RandomString` can be used in the Heat Template to generate a password that can be used when setting up MySQL with this bash script.

## References

<https://gist.github.com/Mins/4602864>

<https://www.digitalocean.com/community/tutorials/how-to-install-linux-nginx-mysql-php-lemp-stack-ubuntu-18-04>

## 14.6 Create A Server Group

### This document is in progress

Server groups can be used to ensure that instances are either placed on the same hypervisor (affinity) or are placed on different hypervisors (anti-affinity).

There are four policies which can be applied to a server group:

- affinity
- soft-affinity
- anti-affinity
- soft-anti-affinity

Server groups can be implemented in a Heat template using the resource OS::Nova::ServerGroup.

### 14.6.1 Syntax

```
resource:
  server_group:
    type: OS::Nova::ServerGroup
    properties:
      name: string #Optional, - Server group name. Any updates cause replacement.
      policies: [string, string] #Optional, a list of string policies to apply.
```

### 14.6.2 Example

```
resources:

  affinity_group:
    type: OS::Nova::ServerGroup
    properties:
      name: hosts on same compute nodes
      policies:
        - affinity

  my_instance:
    type: OS::Nova::Server
    properties:
      image: { get_param: image_id }
      flavor: { get_param: flavor_id }
      key_name: {get_param: KeyName }
      networks:
        - network: Internal #define the network to use as internal
```

(continues on next page)

(continued from previous page)

```

security_groups:
  - { get_param: security_group_id }
user_data_format: RAW
name: server_1 #name for instance
scheduler_hints:
  group: {get_resource: affinity_group}

```

You can list the server groups which are in your project using the command:

```
openstack server group list
```

This should return a table similar to this one:

ID	Name	Policies
6c8030c0-1b33-4470-b26d-51b6cac17bb7	hosts on same compute nodes	affinity

You can also list the members of the server group using:

```
openstack server show <server-group-name/id>
```

For example:

```
openstack server group show 6c8030c0-1b33-4470-b26d-51b6cac17bb7
```

Field	Value
id	6c8030c0-1b33-4470-b26d-51b6cac17bb7
members	87663bdb-c597-4098-b09c-624ec9974572
name	hosts on same compute nodes
policies	affinity

### 14.6.3 References

[https://docs.openstack.org/heat/latest/template\\_guide/openstack.html#OS::Nova::ServerGroup](https://docs.openstack.org/heat/latest/template_guide/openstack.html#OS::Nova::ServerGroup)

<https://docs.syseleven.de/syseleven-stack/en/tutorials/affinity>

[https://docs.openstack.org/heat/latest/template\\_guide/openstack.html#OS::Nova::ServerGroup](https://docs.openstack.org/heat/latest/template_guide/openstack.html#OS::Nova::ServerGroup)

<https://docs.openstack.org/mitaka/config-reference/compute/scheduler.html#servergroupaffinityfilter>

## 14.7 Automating Interactive Scripts

Sometimes there are installations which require user responses to certain questions. One example of this can be found when running the `mysql_secure_installation` command. It is possible to automate these responses when you want to have a database, like MySQL set up and ready to use as root.

To do this, we use **expect**.

According to the linux man page, expect is programmed dialogue which knows what is expected from the program and what the correct response should be.

Typically, Expect is run in a separate script with the first line of the script as:

```
#!/usr/local/bin/expect -<options>
```

However, it is possible to run an expect script within a regular bash script.

An expect script inside a normal bash script would be written as follows:

```
$FOO=(expect -c "
    expect \"<interactive prompt>\"
    send \"<response>\r\"
")
echo "$FOO"
```

### 14.7.1 References

<https://linux.die.net/man/1/expect>

<https://gist.github.com/Mins/4602864>

## 14.8 Accessing MySQL Database from JupyterHub

This document will show how to connect to a MySQL database which is on the same server as JupyterHub.

In MySQL, a new user ‘megan’ has been created with a ‘test’ password. This user has been given access to the database ‘demodb’.

### 14.8.1 Python Packages

#### Python Packages

The following python packages are used in this notebook: - pymysql - sqlalchemy - pandas and scikitlearn - these will be used in one example

## Connecting to a MySQL database

Either pymysql or sqlalchemy can be used to connect to a database, however it is better to use sqlalchemy if you want to import a pandas dataframe as a table into your MySQL database.

```
# pymysql
import pymysql.cursors
#connect to database
connection = pymysql.connect(host='localhost', user='megan', password='test', database=
↳ 'demodb')
```

```
#sqlalchemy
from sqlalchemy import create_engine
eng = create_engine('mysql+pymysql://megan:test@localhost/demodb') #connects to the_
↳ database demodb as user=megan and password=test
eng.connect() #this will show an error if it fails to connect to the database

<sqlalchemy.engine.base.Connection at 0x7f4c5c739f60>
```

After connecting to the database, you can check the list of tables using:

```
#if using sqlalchemy
eng.table_names()

['example', 'iris_table', 'tutorials_tbl']
```

```
#alternatively, pymysql allows you to execute SQL commands:
cursor = connection.cursor()
query = "SELECT * FROM iris_table" #example SQL query
cursor.execute(query) #executes the query, here it states there are 150 entries in the_
↳ table

#output
150
```

## Import dataset into MySQL

The following example will show how to store an Iris dataset as a new table in our demo database.

```
#import python libraries
from sklearn import datasets
import pandas as pd

iris = datasets.load_iris() #load Iris dataset
print(iris.keys())

dict_keys(['data', 'target', 'frame', 'target_names', 'DESCR', 'feature_names', 'filename
↳ '])
```

```
#store dataset in a dataframe
data = pd.DataFrame(iris.data, columns=[iris.feature_names])
print(data) #print the dataframe

#The output would be:
```

	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)
0	5.1	3.5	1.4	0.2
1	4.9	3.0	1.4	0.2
2	4.7	3.2	1.3	0.2
3	4.6	3.1	1.5	0.2
4	5.0	3.6	1.4	0.2
..	...	...	...	...
145	6.7	3.0	5.2	2.3
146	6.3	2.5	5.0	1.9
147	6.5	3.0	5.2	2.0
148	6.2	3.4	5.4	2.3
149	5.9	3.0	5.1	1.8

```
[150 rows x 4 columns]
```

```
#create a new table in MySQL database and import this dataset into the table
#using the engine we have created to connect to the database demodb in MySQL,
#import the pandas dataframe 'data' as a new table in the database called 'iris_table_demo'
data.to_sql(con=eng, name='iris_table_demo')
```

The command above also has an option for the case when the table name matches a table which already exists in the database. This means that you can either replace and overwrite the table in the database, or append the current table in MySQL and add this dataset to the data which is already there.

```
#confirm that the table is in the database:
eng.table_names()

['example', 'iris_table', 'iris_table_demo', 'tutorials_tbl']
```

This shows that we have a new table in the database demodb called 'iris\_table\_demo'.

Let's look at the table iris\_table and print the records from that table.

```
query = "SELECT * FROM iris_table"
cursor.execute(query)
records = cursor.fetchall()
print(records)
```

```
((None, 5.1, 3.5, 1.4, 0.2), (None, 4.9, 3.0, 1.4, 0.2), (None, 4.7, 3.2, 1.3, 0.2),
→(None, 4.6, 3.1, 1.5, 0.2), (None, 5.0, 3.6, 1.4, 0.2), (None, 5.4, 3.9, 1.7, 0.4),
→(None, 4.6, 3.4, 1.4, 0.3), (None, 5.0, 3.4, 1.5, 0.2), (None, 4.4, 2.9, 1.4, 0.2),
→(None, 4.9, 3.1, 1.5, 0.1), (None, 5.4, 3.7, 1.5, 0.2), (None, 4.8, 3.4, 1.6, 0.2),
→(None, 4.8, 3.0, 1.4, 0.1), (None, 4.3, 3.0, 1.1, 0.1), (None, 5.8, 4.0, 1.2, 0.2),
→(None, 5.7, 4.4, 1.5, 0.4), (None, 5.4, 3.9, 1.3, 0.4), (None, 5.1, 3.5, 1.4, 0.3),
→(None, 5.7, 3.8, 1.7, 0.3), (None, 5.1, 3.8, 1.5, 0.3), (None, 5.4, 3.4, 1.7, 0.2),
→(None, 5.1, 3.7, 1.5, 0.4), (None, 4.6, 3.6, 1.0, 0.2), (None, 5.1, 3.3, 1.7, 0.5),
```

(continues on next page)

(continued from previous page)

```

→(None, 4.8, 3.4, 1.9, 0.2), (None, 5.0, 3.0, 1.6, 0.2), (None, 5.0, 3.4, 1.6, 0.4),
→(None, 5.2, 3.5, 1.5, 0.2), (None, 5.2, 3.4, 1.4, 0.2), (None, 4.7, 3.2, 1.6, 0.2),
→(None, 4.8, 3.1, 1.6, 0.2), (None, 5.4, 3.4, 1.5, 0.4), (None, 5.2, 4.1, 1.5, 0.1),
→(None, 5.5, 4.2, 1.4, 0.2), (None, 4.9, 3.1, 1.5, 0.2), (None, 5.0, 3.2, 1.2, 0.2),
→(None, 5.5, 3.5, 1.3, 0.2), (None, 4.9, 3.6, 1.4, 0.1), (None, 4.4, 3.0, 1.3, 0.2),
→(None, 5.1, 3.4, 1.5, 0.2), (None, 5.0, 3.5, 1.3, 0.3), (None, 4.5, 2.3, 1.3, 0.3),
→(None, 4.4, 3.2, 1.3, 0.2), (None, 5.0, 3.5, 1.6, 0.6), (None, 5.1, 3.8, 1.9, 0.4),
→(None, 4.8, 3.0, 1.4, 0.3), (None, 5.1, 3.8, 1.6, 0.2), (None, 4.6, 3.2, 1.4, 0.2),
→(None, 5.3, 3.7, 1.5, 0.2), (None, 5.0, 3.3, 1.4, 0.2), (None, 7.0, 3.2, 4.7, 1.4),
→(None, 6.4, 3.2, 4.5, 1.5), (None, 6.9, 3.1, 4.9, 1.5), (None, 5.5, 2.3, 4.0, 1.3),
→(None, 6.5, 2.8, 4.6, 1.5), (None, 5.7, 2.8, 4.5, 1.3), (None, 6.3, 3.3, 4.7, 1.6),
→(None, 4.9, 2.4, 3.3, 1.0), (None, 6.6, 2.9, 4.6, 1.3), (None, 5.2, 2.7, 3.9, 1.4),
→(None, 5.0, 2.0, 3.5, 1.0), (None, 5.9, 3.0, 4.2, 1.5), (None, 6.0, 2.2, 4.0, 1.0),
→(None, 6.1, 2.9, 4.7, 1.4), (None, 5.6, 2.9, 3.6, 1.3), (None, 6.7, 3.1, 4.4, 1.4),
→(None, 5.6, 3.0, 4.5, 1.5), (None, 5.8, 2.7, 4.1, 1.0), (None, 6.2, 2.2, 4.5, 1.5),
→(None, 5.6, 2.5, 3.9, 1.1), (None, 5.9, 3.2, 4.8, 1.8), (None, 6.1, 2.8, 4.0, 1.3),
→(None, 6.3, 2.5, 4.9, 1.5), (None, 6.1, 2.8, 4.7, 1.2), (None, 6.4, 2.9, 4.3, 1.3),
→(None, 6.6, 3.0, 4.4, 1.4), (None, 6.8, 2.8, 4.8, 1.4), (None, 6.7, 3.0, 5.0, 1.7),
→(None, 6.0, 2.9, 4.5, 1.5), (None, 5.7, 2.6, 3.5, 1.0), (None, 5.5, 2.4, 3.8, 1.1),
→(None, 5.5, 2.4, 3.7, 1.0), (None, 5.8, 2.7, 3.9, 1.2), (None, 6.0, 2.7, 5.1, 1.6),
→(None, 5.4, 3.0, 4.5, 1.5), (None, 6.0, 3.4, 4.5, 1.6), (None, 6.7, 3.1, 4.7, 1.5),
→(None, 6.3, 2.3, 4.4, 1.3), (None, 5.6, 3.0, 4.1, 1.3), (None, 5.5, 2.5, 4.0, 1.3),
→(None, 5.5, 2.6, 4.4, 1.2), (None, 6.1, 3.0, 4.6, 1.4), (None, 5.8, 2.6, 4.0, 1.2),
→(None, 5.0, 2.3, 3.3, 1.0), (None, 5.6, 2.7, 4.2, 1.3), (None, 5.7, 3.0, 4.2, 1.2),
→(None, 5.7, 2.9, 4.2, 1.3), (None, 6.2, 2.9, 4.3, 1.3), (None, 5.1, 2.5, 3.0, 1.1),
→(None, 5.7, 2.8, 4.1, 1.3), (None, 6.3, 3.3, 6.0, 2.5), (None, 5.8, 2.7, 5.1, 1.9),
→(None, 7.1, 3.0, 5.9, 2.1), (None, 6.3, 2.9, 5.6, 1.8), (None, 6.5, 3.0, 5.8, 2.2),
→(None, 7.6, 3.0, 6.6, 2.1), (None, 4.9, 2.5, 4.5, 1.7), (None, 7.3, 2.9, 6.3, 1.8),
→(None, 6.7, 2.5, 5.8, 1.8), (None, 7.2, 3.6, 6.1, 2.5), (None, 6.5, 3.2, 5.1, 2.0),
→(None, 6.4, 2.7, 5.3, 1.9), (None, 6.8, 3.0, 5.5, 2.1), (None, 5.7, 2.5, 5.0, 2.0),
→(None, 5.8, 2.8, 5.1, 2.4), (None, 6.4, 3.2, 5.3, 2.3), (None, 6.5, 3.0, 5.5, 1.8),
→(None, 7.7, 3.8, 6.7, 2.2), (None, 7.7, 2.6, 6.9, 2.3), (None, 6.0, 2.2, 5.0, 1.5),
→(None, 6.9, 3.2, 5.7, 2.3), (None, 5.6, 2.8, 4.9, 2.0), (None, 7.7, 2.8, 6.7, 2.0),
→(None, 6.3, 2.7, 4.9, 1.8), (None, 6.7, 3.3, 5.7, 2.1), (None, 7.2, 3.2, 6.0, 1.8),
→(None, 6.2, 2.8, 4.8, 1.8), (None, 6.1, 3.0, 4.9, 1.8), (None, 6.4, 2.8, 5.6, 2.1),
→(None, 7.2, 3.0, 5.8, 1.6), (None, 7.4, 2.8, 6.1, 1.9), (None, 7.9, 3.8, 6.4, 2.0),
→(None, 6.4, 2.8, 5.6, 2.2), (None, 6.3, 2.8, 5.1, 1.5), (None, 6.1, 2.6, 5.6, 1.4),
→(None, 7.7, 3.0, 6.1, 2.3), (None, 6.3, 3.4, 5.6, 2.4), (None, 6.4, 3.1, 5.5, 1.8),
→(None, 6.0, 3.0, 4.8, 1.8), (None, 6.9, 3.1, 5.4, 2.1), (None, 6.7, 3.1, 5.6, 2.4),
→(None, 6.9, 3.1, 5.1, 2.3), (None, 5.8, 2.7, 5.1, 1.9), (None, 6.8, 3.2, 5.9, 2.3),
→(None, 6.7, 3.3, 5.7, 2.5), (None, 6.7, 3.0, 5.2, 2.3), (None, 6.3, 2.5, 5.0, 1.9),
→(None, 6.5, 3.0, 5.2, 2.0), (None, 6.2, 3.4, 5.4, 2.3), (None, 5.9, 3.0, 5.1, 1.8))

```

However, it is better to first import the table into a pandas data frame:

```
df = pd.read_sql('SELECT * FROM iris_table', con=connection)
```

```
print(df)
```

```

index ('sepal length (cm)',) ('sepal width (cm)',) \
0      None                5.1                3.5
1      None                4.9                3.0

```

(continues on next page)

(continued from previous page)

```

2      None      4.7      3.2
3      None      4.6      3.1
4      None      5.0      3.6
..     ...      ...      ...
145    None      6.7      3.0
146    None      6.3      2.5
147    None      6.5      3.0
148    None      6.2      3.4
149    None      5.9      3.0

('petal length (cm)',) ('petal width (cm)',)
0      1.4      0.2
1      1.4      0.2
2      1.3      0.2
3      1.5      0.2
4      1.4      0.2
..     ...      ...
145    5.2      2.3
146    5.0      1.9
147    5.2      2.0
148    5.4      2.3
149    5.1      1.8

[150 rows x 5 columns]
```

## 14.8.2 References

<https://overiq.com/sqlalchemy-101/installing-sqlalchemy-and-connecting-to-database/>

<https://pandas.pydata.org/pandas-docs/stable/reference/index.html>

[https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.to\\_sql.html](https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.to_sql.html)

<https://towardsdatascience.com/sqlalchemy-python-tutorial-79a577141a91>

<https://linuxize.com/post/show-tables-in-mysql-database/>

<https://pynative.com/python-mysql-select-query-to-fetch-data/>

## 14.8.3 MySQL

A list of useful MySQL commands can be found here:

<http://g2pc1.bu.edu/~qzpeng/manual/MySQL%20Commands.htm>



## 15.1 Manila: Shared File Systems as a Service

OpenStack Manila provides services for managing shared file systems and is the third storage OpenStack component alongside Cinder and Swift. Manila provides a way to provision remote, sharable file systems (shares) which can be accessed by multiple instances simultaneously and file systems can have access rules assigned to control read-write and read-only access to the file system. On STFC Cloud, Manila provides a way to create **Ceph file systems** (CephFS).

### 15.1.1 Manila Commands

In OpenStack Train, the Manila CLI is mainly used for handling shares. There are some limited commands in the OpenStack CLI for shares.

---

**Note:** To use Manila commands in OpenStack Train, the compatible version of the python package `python-manilaclient` needs to be installed. For OpenStack Train, this is `v2.0.0`. This can be installed using `pip install python-manilaclient=2.0.0`.

---

#### OpenStack CLI

In the OpenStack CLI the following commands are available:

```
openstack share --help

share create # create a new share
share delete # delete a share
share list  # list all shares
share show  # view details about a share
```

## Manila CLI

Manila provides a CLI for not only for managing shares, but also for viewing the share types available, managing access rules for a share, and viewing the quota for share in your project. Commands are run using `manila <sub-command>`. The next section will demonstrate how to do some of the Manila commands.

### 15.1.2 Usage

#### View Share Quota

You can view either the overall quota using `manila quota-show`, or the quota and the total quota used with `manila absolute-limits`.

```
$ manila quota-show
+-----+-----+
| Property          | Value          |
+-----+-----+
| gigabytes         | 250            |
| id                | PROJECT_ID     |
| share_group_snapshots | 0             |
| share_groups      | 5             |
| share_networks    | 0             |
| shares            | 10            |
| snapshot_gigabytes | 0             |
| snapshots         | 0             |
+-----+-----+
```

```
$ manila absolute-limits
+-----+-----+
| Name              | Value |
+-----+-----+
| maxTotalShareGigabytes | 250   |
| maxTotalShareNetworks  | 0     |
| maxTotalShareSnapshots | 0     |
| maxTotalShares         | 10    |
| maxTotalSnapshotGigabytes | 250   |
| totalShareGigabytesUsed | 3     |
| totalShareNetworksUsed  | 0     |
| totalShareSnapshotsUsed | 0     |
| totalSharesUsed         | 3     |
| totalSnapshotGigabytesUsed | 0     |
+-----+-----+
```

## List Share Types Available

Share types available can be listed using `manila share-type list` or `openstack share list`. This will list the share types currently available and supported.

```
$ openstack share list
+-----+-----+-----+-----+-----+-----+-----+
| ID | Name | Size | Share Proto |
+-----+-----+-----+-----+-----+-----+-----+
| a6b215a6-00c5-46a5-b1db-d86559097896 | test_share | 1 | CEPHFS |
+-----+-----+-----+-----+-----+-----+-----+
| 7a1beb23-8dee-4709-9bcd-c947ae006653 | updated_demo_share | 1 | CEPHFS |
+-----+-----+-----+-----+-----+-----+-----+
| available | False | cephfs | None |
| available | False | cephfs | nova |
```

## Create a Share

Shares can be created by users using the `manila create` command. The required arguments are the share protocol and the size of the share in GiB.

```
$ manila create --help
usage: manila create [--snapshot-id <snapshot-id>] [--name <name>] [--metadata [
<key=value> [<key=value> ...]]] [--share-network <network-info>] [--description
<description>] [--share-type <share-type>]
[--public] [--availability-zone <availability-zone>] [--share-group
<share_group>]
<share_protocol> <size>

Creates a new share (NFS, CIFS, CephFS, GlusterFS, HDFS or MAPRFS).

Positional arguments:
<share_protocol>  Share protocol (NFS, CIFS, CephFS, GlusterFS, HDFS or MAPRFS).
<size>            Share size in GiB.

Optional arguments:
--snapshot-id <snapshot-id>, --snapshot_id <snapshot-id>
                        Optional snapshot ID to create the share from. (Default=None)
--name <name>          Optional share name. (Default=None)
--metadata [<key=value> [<key=value> ...]]
                        Metadata key=value pairs (Optional, Default=None).
--share-network <network-info>, --share_network <network-info>
                        Optional network info ID or name.
--description <description>
                        Optional share description. (Default=None)
--share-type <share-type>, --share_type <share-type>, --volume-type <share-type>, --
volume_type <share-type>
                        Optional share type. Use of optional volume type is deprecated.
(Default=None)
```

(continues on next page)

(continued from previous page)

```
--public          Level of visibility for share. Defines whether other tenants are
↳able to see it or not. (Default=False)
--availability-zone <availability-zone>, --availability_zone <availability-zone>, --az
↳<availability-zone>
                    Availability zone in which share should be created.
--share-group <share-group>, --share_group <share-group>, --group <share-group>
                    Optional share group name or ID in which to create the share.
↳(Experimental, Default=None).
```

So if we want to create a new CephFS share of size 1GiB, we can use the following command:

```
$ manila create --name demo_share --description "Demo creating a share" --share-type
↳SHARE_TYPE_ID CephFS 1
+-----+-----+
| Property                                | Value                                |
|-----+-----+-----+
| status                                | creating                             |
| share_type_name                        | cephfs                              |
| description                            | Demo creating a share                |
| availability_zone                       | None                                 |
| share_network_id                       | None                                 |
| share_group_id                         | None                                 |
| revert_to_snapshot_support             | False                               |
| access_rules_status                    | active                              |
| snapshot_id                            | None                                 |
| create_share_from_snapshot_support     | False                               |
| is_public                              | False                               |
| task_state                             | None                                 |
| snapshot_support                       | False                               |
| id                                     | 7a1beb23-8dee-4709-9bcd-c947ae006653 |
| size                                  | 1                                   |
| source_share_group_snapshot_member_id | None                                 |
| user_id                                | USER_ID                             |
```

(continues on next page)

(continued from previous page)

↩			
↩	name	demo_share	↪
↩			
↩	share_type	SHARE_TYPE_ID	↪
↩			
↩	has_replicas	False	↪
↩			
↩	replication_type	None	↪
↩			
↩	created_at	2022-10-19T15:03:28.000000	↪
↩			
↩	share_proto	CEPHFS	↪
↩			
↩	mount_snapshot_support	False	↪
↩			
↩	project_id	PROJECT_ID	↪
↩			
↩	metadata	{}	↪
↩			
↩	+-----+	+-----+	
↩	↩-----↪		

## Update a Share

Once a share has been created in Manila, there are only three properties which can be updated: - Name of the share. - The description for the share. - Change the visibility of the share to public or private.

```
$ manila update --help
usage: manila update [--name <name>] [--description <description>] [--is-public <is_
  ↵ public>] <share>
```

Rename a share.

Positional arguments:

<share>	Name or ID of the share to rename.
---------	------------------------------------

Optional arguments:

```
--name <name>           New name for the share.
--description <description>
                        Optional share description. (Default=None)
--is-public <is_public>, --is_public <is_public>
                        Public share is visible for all tenants.
```

For example, we can update the name of a share from `demo_share` to `updated_demo_share` in the following way:

```
$ manila update --name updated_demo_share demo_share
```

Then we can see the updated share in the list of shares in the current project:

```
$ manila list
```

(continues on next page)

(continued from previous page)

```

┌───┴───┐
| ID | Name | Size | Share Proto |
├───┴───┤
| Status | Is Public | Share Type | Name | Host | Availability | Zone |
├───┴───┤
┌───┴───┐
| 7a1beb23-8dee-4709-9bcd-c947ae006653 | updated_demo_share | 1 | CEPHFS |
├───┴───┤
| available | False | cephfs | | nova |
├───┴───┤
┌───┴───┐

```

## Extend a Share

The size of a share can be increased using the `manila extend` command.

```
$ manila extend --help
usage: manila extend <share> <new_size>
```

Increases the size of an existing share.

Positional arguments:

<code>&lt;share&gt;</code>	Name or ID of share to extend.
<code>&lt;new_size&gt;</code>	New size of share, <b>in</b> GiBs.

For example, if we want to extend a demo share from 1GiBs to 2GiBs, we can do the following:

```
$ manila extend updated_demo_share 2
```

Viewing the details of the share we can see that the size of the share has been updated.

```
$ manila show updated_demo_share
```

Property	Value
status	available
share_type_name	cephfs
description	Demo creating a share
availability_zone	nova
share_network_id	None
share_group_id	None
revert_to_snapshot_support	False
access_rules_status	active

(continues on next page)

(continued from previous page)

↩	snapshot_id	None		↪
↩	create_share_from_snapshot_support	False		↪
↩	is_public	False		↪
↩	task_state	None		↪
↩	snapshot_support	False		↪
↩	id	7a1beb23-8dee-4709-9bcd-c947ae006653		↪
↩	size	2		↪
↩	source_share_group_snapshot_member_id	None		↪
↩	user_id	USER_ID		↪
↩	name	updated_demo_share		↪
↩	share_type	SHARE_TYPE		↪
↩	has_replicas	False		↪
↩	replication_type	None		↪
↩	created_at	2022-10-19T15:03:28.000000		↪
↩	share_proto	CEPHFS		↪
↩	mount_snapshot_support	False		↪
↩	project_id	PROJECT_ID		↪
↩	metadata	{}		↪
↩	export_locations			↪
↩		path = EXPORT_PATH		↪
↩		id = EXPORT_LOCATIONS_ID		↪
↩		preferred = False		↪
↩	+	+	+	↪
↩	+	+	+	↪

Shrink a Share

**Warning:** This can only be done through the command line only.

The size of a share can be reduced using the `manila shrink` command.

```
$ manila shrink --help
usage: manila shrink <share> <new_size>

Decreases the size of an existing share.

Positional arguments:
  <share>      Name or ID of share to shrink.
  <new_size>   New size of share, in GiBs.
```

Using the example in the previous section, we can reduce the size of a share from 2GiB to 1Gib using:

```
$ manila shrink updated_demo_share 1
```

We can see using `manila show updated_demo_share` that the size of the share has been updated:

```
$ manila show updated_demo_share
+-----+-----+
| Property | Value |
+-----+-----+
| status | available |
| share_type_name | cephfs |
| description | Demo creating a share |
| availability_zone | nova |
| share_network_id | None |
| share_group_id | None |
| revert_to_snapshot_support | False |
| access_rules_status | active |
| snapshot_id | None |
| create_share_from_snapshot_support | False |
| is_public | False |
| task_state | None |
```

(continues on next page)

(continued from previous page)

snapshot_support	False		
id	7a1beb23-8dee-4709-9bcd-c947ae006653		
size	1		
source_share_group_snapshot_member_id	None		
user_id	USER_ID		
name	updated_demo_share		
share_type	SHARE_TYPE		
has_replicas	False		
replication_type	None		
created_at	2022-10-19T15:03:28.000000		
share_proto	CEPHFS		
mount_snapshot_support	False		
project_id	PROJECT_ID		
metadata	{}		
export_locations			
	path = EXPORT_PATH		
	id = EXPORT_LOCATIONS_ID		
	preferred = False		
+-----+			
+-----+			

## Add Access Rule through CLI

Access rules can be created for a share through the Web UI or on the command line. You can allow either read-write (RW) or read-only (RO) access to the share.

**Warning:** Only cephx access rules can be used for shares as only CephFS shares are currently supported. Any other type of access rule created will go into error state.

To create a new access rule for a share, e.g. a share named `demo_share`, we can use the `manila access-allow` command:

```
$ manila access-allow --help
usage: manila access-allow [--access-level <access_level>] [--metadata [<key=value> [
↳<key=value> ...]]] <share> <access_type> <access_to>

Allow access to a given share.

Positional arguments:
  <share>          Name or ID of the NAS share to modify.
  <access_type>    Access rule type (only "ip", "user"(user or group), "cert" or
↳"cephx" are supported).
  <access_to>      Value that defines access.

Optional arguments:
  --access-level <access_level>, --access_level <access_level>
                                  Share access level ("rw" and "ro" access levels are supported).
↳Defaults to rw.
  --metadata [<key=value> [<key=value> ...]]
                                  Space Separated list of key=value pairs of metadata items.
↳OPTIONAL: Default=None.

$ manila access-allow demo_share cephx alice
```

**Note:** Here, `access_to` refers to the name given to the cephx access rule. This could be the name of a user to give share access to for example. The value given for `access_to` is used in later examples demonstrating how to mount a share.

Then we can view the access rules for the share using:

```
$ manila access-list demo_share
+-----+-----+-----+-----+
↳+-----+-----+-----+-----+
↳+-----+
| id                  | access_type | access_to | access_level |
↳state | access_key          | created_at |               |
↳updated_at          |
+-----+-----+-----+-----+
↳+-----+-----+-----+-----+
↳+-----+
| 56907a0c-024a-465e-8ebc-5a0b085ac87b | cephx      | alice     | rw           |
↳active | ACCESS_KEY          | 2022-10-14T14:55:59.000000 | 2022-
↳10-14T14:55:59.000000 |
+-----+-----+-----+-----+
↳+-----+-----+-----+-----+
↳+-----+

```

## Remove Access Rule through CLI

Access rules can be removed from a share using `manila access-deny`:

```
$ manila access-deny --help
usage: manila access-deny <share> <id>

Deny access to a share.

Positional arguments:
  <share>  Name or ID of the NAS share to modify.
  <id>     ID of the access rule to be deleted.
```

## Delete a Share

A share can be deleted by using the `manila delete` command:

```
$ manila delete --help
usage: manila delete [--share-group <share-group>] <share> [<share> ...]

Remove one or more shares.

Positional arguments:
  <share>          Name or ID of the share(s).

Optional arguments:
  --share-group <share-group>, --share_group <share-group>, --group <share-group>
                                Optional share group name or ID which contains the share.
  ↪ (Experimental, Default=None).
```

### 15.1.3 How to access a CephFS Share

There are two methods that can be used to mount a share onto a VM:

- Kernel method
- Ceph FUSE method

---

**Note:** The kernel method is preferred as it uses significantly less CPU marshalling data through userspace.

---

The following examples assume that the following have been set up:

- VM (examples use an Ubuntu VM)
- CephFS Share and have the export path given in the share details copied to use
- cephx share access rules

## Access a share using the Kernel Client

The kernel client requires the `ceph-common` package to be installed. To access the filesystem on a share, we can use the `mount` command of the form:

```
# breakdown of the command

mount -t ceph {mon1 ip addr}:6789,{mon2 ip addr}:6789,{mon3 ip addr}:6789:/{mount-point} \
↪ # export path for the share

-o name={access-rule-name},secret={access-key} # name of the access rule and the secret \
↪ associated to it

{mount-path} # path you want to attach the share to
```

The directory will be owned by `root` by default. To allow another user to read and write add the appropriate user and group like so:

```
chown user:user {mount-path}
```

## Access a share using the FUSE Client

In order to use the FUSE client, make sure the version of the `ceph-fuse` package is at least `octopus`. The version of the package can be found using:

```
~$ ceph-fuse --version
ceph version 15.2.16 (d46a73d6d0a67a79558054a3a5a72cb561724974) octopus (stable)
```

To use the FUSE client, we need to set up a keyring and a `ceph.conf` file. If we have named the `cephx` rule created with the name `alice`, then the keyring file will need to be named `alice.keyring` and needs to contain the following:

```
[client.alice]
    key = CEPHX_KEY
```

Where `CEPHX_KEY` is the secret for the specific access rule.

Then for the `ceph.conf` file, we need to add the IP addresses for the monitors from the ceph cluster. The IP address for the ceph monitors for the share can be found using `manila show <share-id>` under the `export locations` field.

```
[client]
    client quota = True
    mon host = MON_HOST_IP_ADDRESSES
```

Then the filesystem can be mounted to a test directory using the `alice` `cephx` access rule:

```
sudo ceph-fuse ~/mnt \
--id=alice \
--conf=./ceph.conf \
--keyring=./alice.keyring \
--client-mountpoint={mount-point}
```

Here the mount point is the path to the share on the ceph cluster. The share path can be found using `manila show <share-id>` under the `export locations` field.

### 15.1.4 References:

<https://docs.openstack.org/manila/train/user/create-and-manage-shares.html>

<https://docs.openstack.org/manila/train/admin/shared-file-systems-crud-share.html#manage-access-to-share>

[https://docs.openstack.org/manila/train/admin/cephfs\\_driver.html](https://docs.openstack.org/manila/train/admin/cephfs_driver.html)

## 15.2 Using Shares on Kubernetes Clusters

**Warning:** OpenStack Manila is currently in technical preview on STFC Cloud. If you have any feedback or suggestions, please sent it to [cloud-support@stfc.ac.uk](mailto:cloud-support@stfc.ac.uk)

This feature for using Shares with Kubernetes is also currently being tested by the Cloud Team.

To use shares on Kubernetes clusters we need the following **csi** installed and deployed onto the cluster:

- **manila-csi:** <https://github.com/kubernetes/cloud-provider-openstack/blob/master/docs/manila-csi-plugin/using-manila-csi-plugin.md>
- **ceph-csi:** <https://github.com/ceph/ceph-csi>

The **manila-csi** is designed to be lightweight and only handles the interaction between Kubernetes and OpenStack to create, delete, and access shares. The **ceph-csi** handles the CephFS side of the filesystem on the Kubernetes cluster. See <https://github.com/kubernetes/cloud-provider-openstack/blob/master/docs/manila-csi-plugin/developers-csi-manila.md#notes-on-design> for more information on how **manila-csi** is designed.

### 15.2.1 Deploying onto the Cluster

Both the **manila-csi** and **ceph-csi** can be deployed to a Kubernetes cluster using a Helm chart.

#### ceph-csi

The **ceph-csi** can be deployed in the following way:

```
# Add the chart repository
helm repo add ceph-csi https://ceph.github.io/csi-charts
# Create namespace for the chart
kubectl create namespace ceph-csi-cephfs
# Install the chart
helm install --namespace "ceph-csi-cephfs" "ceph-csi-cephfs" ceph-csi/ceph-csi-cephfs
```

You should then be able to see the following pods in the *ceph-csi-cephfs* namespace:

```
$ kubectl get pods -n ceph-csi-cephfs
```

NAME	READY	STATUS	RESTARTS	AGE
ceph-csi-cephfs-nodeplugin-6554g	3/3	Running	0	17d
ceph-csi-cephfs-nodeplugin-6xcfc	3/3	Running	0	17d
ceph-csi-cephfs-nodeplugin-ddt84	3/3	Running	0	17d
ceph-csi-cephfs-nodeplugin-h97j5	3/3	Running	0	17d
ceph-csi-cephfs-nodeplugin-hxj95	3/3	Running	0	17d

(continues on next page)

(continued from previous page)

ceph-csi-cephfs-nodeplugin-j65fp	3/3	Running	0	17d
ceph-csi-cephfs-nodeplugin-ltt7q	3/3	Running	0	17d
ceph-csi-cephfs-nodeplugin-mhwr8	3/3	Running	0	17d
ceph-csi-cephfs-nodeplugin-wrjsk	3/3	Running	0	17d
ceph-csi-cephfs-nodeplugin-zfl8w	3/3	Running	0	17d
ceph-csi-cephfs-provisioner-765854df4c-csdft	5/5	Running	0	17d
ceph-csi-cephfs-provisioner-765854df4c-nzp77	5/5	Running	0	17d
ceph-csi-cephfs-provisioner-765854df4c-qznz4	5/5	Running	0	17d

## manila-csi

Before deploying the Helm chart for manila-csi, we need to create a `values.yaml` file in order to deploy the csi to support CephFS shares:

```
# manila-csi-values.yaml
nameOverride: ""
fullNameOverride: ""

# Base name of the CSI Manila driver
driverName: manila.csi.openstack.org

# Enabled Manila share protocols
shareProtocols:
- protocolSelector: CEPHFS
fsGroupPolicy: None
fwdNodePluginEndpoint:
  dir: /var/lib/kubelet/plugins/cephfs.csi.ceph.com
  sockFile: csi.sock
```

Then we can deploy the chart:

```
# add helm repo
helm repo add cpo https://kubernetes.github.io/cloud-provider-openstack
helm repo update

# install helm chart with values.yaml file 'manila-csi-values.yaml'
helm install manila-csi -f manila-csi-values.yaml cpo/openstack-manila-csi
```

Then you should see pods similar to the following being created onto the cluster (for this example the chart has been deployed into the default namespace):

```
$ kubectl get pod
```

NAME	READY	STATUS	
↪ RESTARTS    AGE			
manila-csi-openstack-manila-csi-controllerplugin-0	4/4	Running	0
↪            14d			
manila-csi-openstack-manila-csi-nodeplugin-6plbt	2/2	Running	0
↪            14d			
manila-csi-openstack-manila-csi-nodeplugin-9dxrl	2/2	Running	0
↪            14d			
manila-csi-openstack-manila-csi-nodeplugin-cjhvp	2/2	Running	0

(continues on next page)

(continued from previous page)

↪ 14d				
manila-csi-openstack-manila-csi-nodeplugin-f72s9	2/2	Running	0	⬇
↪ 14d				
manila-csi-openstack-manila-csi-nodeplugin-ghfmx	2/2	Running	0	⬇
↪ 14d				
manila-csi-openstack-manila-csi-nodeplugin-h99sx	2/2	Running	0	⬇
↪ 14d				
manila-csi-openstack-manila-csi-nodeplugin-k5g9k	2/2	Running	0	⬇
↪ 14d				
manila-csi-openstack-manila-csi-nodeplugin-mlvpp	2/2	Running	0	⬇
↪ 14d				
manila-csi-openstack-manila-csi-nodeplugin-qpd9m	2/2	Running	0	⬇
↪ 14d				
manila-csi-openstack-manila-csi-nodeplugin-sltkx	2/2	Running	0	⬇
↪ 14d				

Next step is to manually create the secret for the manila-csi to use in order to interact with OpenStack.

**Note:** This is close but not identical to the contents of a `clouds.yaml` secret, we are planning on providing a fix upstream to bring them into line.

The secrets template looks similar to this:

```
# manila_csi_secret.yaml
apiVersion: v1
kind: Secret
metadata:
  name: csi-manila-secret
  namespace: kube-system # or another namespace - pvc will require the namespace where the
↪ secret is stored
StringData:
  # mandatory
  os-authURL: "<auth-url>"
  os-region: "<region>"
  # authentication using application credentials
  os-applicationCredentialID: "<application-credential-id>"
  os-applicationCredentialSecret: "<application-credential-secret>"
  # authentication using user credentials
  os-projectID: "<project-id>"
  os-username: "<username>"
  os-password: "<password>"
  os-domainName: "<domain>"
```

**Warning:** Although creating the secret using a service account credential has been tested, it is not recommended to use when preparing the secret for the manila-csi on Kubernetes. Ideally an application credential should be used.

The secret can then be created on the cluster using:

```
kubectl apply -f manila_csi_secret.yaml
```

## 15.2.2 Examples: Provisioning Manila Shares onto the Cluster

**Note:** The following examples are following the examples from <https://github.com/kubernetes/cloud-provider-openstack/tree/master/examples/manila-csi-plugin>

### Static Share Provisioning

**Note:** If a default storage class has been set up in the cluster, then `storageClass: ""` needs to be included in the `pvc.yaml` file. Otherwise, the cluster will default to attempting to make a pvc based on the cluster's default storage class.

- Create a persistent volume claim (PVC) and persistent volume:

```
apiVersion: v1
kind: PersistentVolume
metadata:
  name: preprovisioned-cephfs-share
  labels:
    name: preprovisioned-cephfs-share
spec:
  accessModes:
    - ReadWriteMany
  capacity:
    storage: 1Gi
  csi:
    driver: cephfs.manila.csi.openstack.org
    volumeHandle: preprovisioned-cephfs-share
    nodeStageSecretRef:
      name: csi-manila-secret
      namespace: kube-system
    nodePublishSecretRef:
      name: csi-manila-secret
      namespace: kube-system
    volumeAttributes:
      shareID: <share-id> # the share ID of the share you want to use as the persistent
↪ volume in the cluster
      shareAccessID: <access-rule-id> # the ID of a cephx access rule created for the
↪ share.
---
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: existing-cephfs-share-pvc
spec:
  accessModes:
    - ReadWriteMany
  storageClassName: "" # to override any default storageclass in cluster
resources:
  requests:
```

(continues on next page)

(continued from previous page)

```

    storage: 1Gi
  selector:
    matchExpressions:
      - key: name
        operator: In
        values: ["preprovisioned-cephfs-share"]

```

**Note:** The ID of access rules for a specific share can be found using `manila access-list <share-name/ID>`. This will list all the access rules for a given share and include the ID for each rule. Access rules can be managed in the web UI or using the CLI. For more information on how to create a access rule using the CLI, see <https://stfc-cloud-docs.readthedocs.io/en/latest/Manila/usage.html#add-access-rule-through-cli>

You should be able to create the persistent volume claim created successfully here:

```

$ kubectl apply -f pvc_pv.yaml

$ kubectl get pvc

NAME                                STATUS    VOLUME                                CAPACITY   ACCESS MODES   STORAGECLASS   AGE
existing-cephfs-share-pvc           Bound     preprovisioned-cephfs-share          1Gi        RWX            cephfs         19h

$ kubectl describe pvc existing-cephfs-share-pvc

Name:          existing-cephfs-share-pvc
Namespace:     default
StorageClass:
Status:        Bound
Volume:        preprovisioned-cephfs-share
Labels:        <none>
Annotations:   pv.kubernetes.io/bind-completed: yes
               pv.kubernetes.io/bound-by-controller: yes
Finalizers:    [kubernetes.io/pvc-protection]
Capacity:      1Gi
Access Modes:  RWX
VolumeMode:    Filesystem
Used By:       existing-cephfs-share-pod
Events:        <none>

$ kubectl get pv

NAME                                CAPACITY   ACCESS MODES   RECLAIM POLICY   STATUS    CLAIM                                STORAGECLASS   AGE
preprovisioned-cephfs-share          1Gi        RWX            Retain            Bound     default/existing-cephfs-share-pvc   cephfs         19h

$ kubectl describe pv preprovisioned-cephfs-share

Name:          preprovisioned-cephfs-share
Labels:        name=preprovisioned-cephfs-share

```

(continues on next page)

(continued from previous page)

```

Annotations:    pv.kubernetes.io/bound-by-controller: yes
Finalizers:     [kubernetes.io/pv-protection]
StorageClass:
Status:         Bound
Claim:          default/existing-cephfs-share-pvc
Reclaim Policy: Retain
Access Modes:   RWX
VolumeMode:     Filesystem
Capacity:       1Gi
Node Affinity:  <none>
Message:
Source:
  Type:          CSI (a Container Storage Interface (CSI) volume source)
  Driver:         cephfs.manila.csi.openstack.org
  FSType:         cephfs
  VolumeHandle:   preprovisioned-cephfs-share
  ReadOnly:       false
  VolumeAttributes: shareAccessID=SHARE_ACCESS_RULE_ID
                   shareID=SHARE_ID
Events:         <none>

```

## Dynamic Share Provisioning

Shares can also be created on demand and attached to specific pods.

First, a storage class needs to be created:

```

apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: csi-manila-cephfs
provisioner: cephfs.manila.csi.openstack.org
allowVolumeExpansion: true
parameters:
  # Manila share type
  type: cephfs

csi.storage.k8s.io/provisioner-secret-name: csi-manila-secret
csi.storage.k8s.io/provisioner-secret-namespace: kube-system
csi.storage.k8s.io/controller-expand-secret-name: csi-manila-secret
csi.storage.k8s.io/controller-expand-secret-namespace: kube-system
csi.storage.k8s.io/node-stage-secret-name: csi-manila-secret
csi.storage.k8s.io/node-stage-secret-namespace: kube-system
csi.storage.k8s.io/node-publish-secret-name: csi-manila-secret
csi.storage.k8s.io/node-publish-secret-namespace: kube-system

```

Then the storage class can be created from this template using:

```
kubectl apply -f storageclass.yaml
```

Next, a persistent volume claim needs to be created in order to define the size of the share and the access mode.

```

apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: new-cephfs-share-pvc
spec:
  accessModes:
    - ReadWriteMany
  resources:
    requests:
      storage: 1Gi
    storageClassName: csi-manila-cephfs

```

To create the pvc:

```
kubectl apply -f dynamic_pvc.yaml
```

Then, for example, if we want to have an nginx pod spun up with a new share attached we can use the following template:

```

apiVersion: v1
kind: Pod
metadata:
  name: new-cephfs-share-pod
spec:
  containers:
    - name: web-server
      image: nginx
      imagePullPolicy: IfNotPresent
      volumeMounts:
        - name: mypvc
          mountPath: /var/lib/www
  volumes:
    - name: mypvc
      persistentVolumeClaim:
        claimName: new-cephfs-share-pvc
        readOnly: false

```

Then we can create the pod and inspect it:

```

$ kubectl apply -f pod.yaml

$ kubectl describe pod new-cephfs-share-pod
Name:                new-cephfs-share-pod
Namespace:           default
Priority:              0
Service Account:     default
Node:                 cloud-dev-md-nano-fxhmg/10.6.0.204
Start Time:           Thu, 02 Feb 2023 14:21:56 +0000
Labels:               <none>
Annotations:          cni.projectcalico.org/containerID: d0741d0c1f5b8c4733c3a7c090dbaef9e6875508ab074c36cc671bfa51479494
                     cni.projectcalico.org/podIP: 192.168.232.10/32
                     cni.projectcalico.org/podIPs: 192.168.232.10/32
Status:               Running

```

(continues on next page)

(continued from previous page)

```

IP: 192.168.232.10
IPs:
  IP: 192.168.232.10
Containers:
  web-server:
    Container ID: containerd://
↪ 12a76ece022f9cbc45b6d9af3e2c0efa38dafb4734a52dcdbceaa4f63caf174e
    Image: nginx
    Image ID: docker.io/library/
↪ nginx@sha256:b8f2383a95879e1ae064940d9a200f67a6c79e710ed82ac42263397367e7cc4e
    Port: <none>
    Host Port: <none>
    State: Running
      Started: Thu, 02 Feb 2023 14:22:10 +0000
    Ready: True
    Restart Count: 0
    Environment: <none>
    Mounts:
      /var/lib/www from mypvc (rw)
      /var/run/secrets/kubernetes.io/serviceaccount from kube-api-access-6l758 (ro)
Conditions:
  Type          Status
  Initialized    True
  Ready          True
  ContainersReady True
  PodScheduled   True
Volumes:
  mypvc:
    Type: PersistentVolumeClaim (a reference to a PersistentVolumeClaim in the
↪ same namespace)
    ClaimName: new-cephfs-share-pvc
    ReadOnly: false
  kube-api-access-6l758:
    Type: Projected (a volume that contains injected data from
↪ multiple sources)
    TokenExpirationSeconds: 3607
    ConfigMapName: kube-root-ca.crt
    ConfigMapOptional: <nil>
    DownwardAPI: true
QoS Class: BestEffort
Node-Selectors: <none>
Tolerations: node.kubernetes.io/not-ready:NoExecute op=Exists for 300s
              node.kubernetes.io/unreachable:NoExecute op=Exists for 300s
Events: <none>

```

In the OpenStack project where the application credential was created for, we should now see a new share has been created with a cephx access rule.

## 16.1 Octavia

Octavia provides network load balancing for OpenStack and is the reference implementation LBaaS (Load Balancer as a Service). Since the OpenStack Liberty release, Octavia has fully replaced and has become the reference implementation for Load Balancing as a Service (LBaaS v2).

Neutron previously provided reference implementation for LBaaS, however it has since been deprecated and Octavia should be used for creating load balancers. OpenStack strongly advises that users should use Octavia to create Load balancers.

Load balancers can be created through the web UI or the OpenStack CLI.

Octavia works with the OpenStack Projects to provide a service for creating LBaaS pools:

- **Nova:** managing amphora lifecycle and creating compute resources.
- **Neutron:** For network connectivity
- **Keystone:** Authentication against Octavia API
- **Glance:** stores amphora VM image.
- **Oslo:** Communication between Octavia components. Octavia makes extensive use of **Taskflow**.

### 16.1.1 LBaaS Pools

LBaaS pools may consist of the following:

**Load Balancer:** Balances the traffic to server pools. In combination with the health monitor and listener, the load balancer can redistribute traffic to servers in the event of one server failing for example.

**Health Monitor:** Monitors the health of the pool and defines the method for checking the health of each member of the pool.

**Listener:** Represent a listening endpoint for the VIP. Here, the Listener listens for client traffic being directed to the load balanced pool.

**Pool:** A group of servers which are identified by their floating IP addresses. The size of the pool can be increased or decreased in response to traffic to the instances detected by the load balancer.

These can be created through the web UI, command line, or as part of a Heat Stack such as a LAMP stack.

## 16.1.2 Octavia CLI

In order to use the commands provided from Octavia, make sure to you have the following python package installed:

```
pip install python-octaviaclient
```

To test whether the octaviaclient has been installed successfully and we can access the Octavia commands, we can try to list load balancers in our current project:

```
openstack loadbalancer list
```

This should return an empty line if there are no load balancers in your project, or a table containing details of load balancers similar to the table below:

id	vip_address	provisioning_status	provider	project_id
LOADBALANCER_ID	192.168.132.93	ACTIVE	amphora	PROJECT_ID
LOADBALANCER_ID	10.0.0.38	ACTIVE	amphora	PROJECT_ID
LOADBALANCER_ID	10.0.0.254	ACTIVE	amphora	PROJECT_ID

## Commands

Having installed python-octaviaclient, we are now able to run commands for working with load balanced pools.

There are several commands which Octavia provides which we can use for working with loadbalancers, pools, listeners, and health monitors. Below is a list of some of the commands which can be used:

```
#Commands are of the form:
openstack loadbalancer <commands>

#Load Balancers
loadbalancer create # create a load balancer
loadbalancer delete # delete a load balancer
loadbalancer list # list load balancer
loadbalancer show # view details for a load balancer
loadbalancer failover # trigger a load balancer failover
loadbalancer set # update a loadbalancer
loadbalancer stats show # show current stats for the load balancer
loadbalancer status show # show status of the load balancer
loadbalancer unset #clear loadbalancer settings

#Health Monitor
loadbalancer healthmonitor create #create a health monitor
loadbalancer healthmonitor delete #delete a health monitor
```

(continues on next page)

(continued from previous page)

```

loadbalancer healthmonitor list #list health monitors
loadbalancer healthmonitor set #update health monitors
loadbalancer healthmonitor show #view health monitor details
loadbalancer healthmonitor unset #clear health monitor settings

#Listener
loadbalancer listener create #create listener
loadbalancer listener delete #delete listener
loadbalancer listener list #list listeners
loadbalancer listener set #update listener
loadbalancer listener show #view details of a listener
loadbalancer listener stats show #show listener stats
loadbalancer listener unset #clear settings for listener

#Loadbalancer members
loadbalancer member create #create a pool member
loadbalancer member delete #remove pool member
loadbalancer member list #list pool members
loadbalancer member set #update a pool member
loadbalancer member show #view details of a pool member
loadbalancer member unset #clear settings for pool member

#Loadbalanced pools
loadbalancer pool create #create loadbalancer pool
loadbalancer pool delete #delete pool
loadbalancer pool list #list pools
loadbalancer pool set #update pool
loadbalancer pool show #view details of the pool
loadbalancer pool unset #clear pool settings

```

Further commands can be seen using `openstack loadbalancer --help`.

### 16.1.3 Creating LBaaS v2 Pools

LBaaS v2 pools can be created through the Web UI and CLI. Let's look at creating a simple LBaaS v2 pool.

#### Load Balancer

Load balancers can be created using:

```

openstack loadbalancer create [-h] [-f {json,shell,table,value,yaml}] [-c COLUMN] [--
↪noindent] [--prefix PREFIX]
                                [--max-width <integer>] [--fit-width] [--print-
↪empty] [--name <name>]
                                [--description <description>] [--vip-address <vip_
↪address>] [--vip-port-id <vip_port_id>]
                                [--vip-subnet-id <vip_subnet_id>] [--vip-network-id
↪<vip_network_id>]
                                [--vip-qos-policy-id <vip_qos_policy_id>] [--
↪project <project>] [--provider <provider>]
                                [--availability-zone <availability_zone>] [--enable_

```

(continues on next page)

(continued from previous page)

```
↪ | --disable] [--flavor <flavor>] [--wait]
```

optional arguments:

```
-h, --help            show this help message and exit
--name <name>         New load balancer name.
--description <description>
                        Set load balancer description.
--vip-address <vip_address>
                        Set the VIP IP Address.
--vip-qos-policy-id <vip_qos_policy_id>
                        Set QoS policy ID for VIP port. Unset with 'None'.
--project <project>   Project for the load balancer (name or ID).
--provider <provider>
                        Provider name for the load balancer.
--availability-zone <availability_zone>
                        Availability zone for the load balancer.
--enable              Enable load balancer (default).
--disable             Disable load balancer.
--flavor <flavor>     The name or ID of the flavor for the load balancer.
--wait               Wait for action to complete
```

For example, we can create a load balancer that is on a private subnet.

```
openstack loadbalancer create --name test-lb --vip-subnet-id <subnet-id>
```

*#This should return a table similar to:*

Field	Value
admin_state_up	True
availability_zone	
created_at	2020-11-11T17:00:19
description	
flavor_id	None
id	fe22e256-d409-4f91-867f-508d08566892
listeners	
name	test-lb
operating_status	OFFLINE
pools	
project_id	PROJECT_ID
provider	amphora
provisioning_status	PENDING_CREATE
updated_at	None
vip_address	192.168.132.125
vip_network_id	8e748892-365d-4fa9-ac6d-f71f608db340
vip_port_id	f42f085f-734c-43ae-a570-cd2c84f13abc
vip_qos_policy_id	None
vip_subnet_id	d6a914ad-6331-4028-bf8b-3a22f6c20f57

After a few minutes the provisioning status should change from *PENDING\_CREATE* to *ACTIVE*. Then, you can associate a floating IP in order to access the load balancer externally.

To create a load balancer which is to balance incoming traffic from the External network:

```
openstack loadbalancer create --name external-lb --vip-subnet-id External
```

This will create a load balancer and create a floating IP to be associated with it.

## Listener

Listeners are set up and attached to load balancers to ‘listen’ for incoming traffic trying to connect to a pool through the load balancer. Listeners can be created using:

```
openstack loadbalancer listener create [-h] [-f {json,shell,table,value,yaml}] [-c   
↪ COLUMN] [--noindent] [--prefix PREFIX]                                [--max-width <integer>] [--fit-width] [--   
↪ print-empty] [--name <name>]                                         [--description <description>] --protocol   
↪ {TCP,HTTP,HTTPS,TERMINATED_HTTPS,UDP}                                [--connection-limit <limit>] [--default-   
↪ pool <pool>]                                                         [--default-tls-container-ref <container_   
↪ ref>]                                                                  [--sni-container-refs [<container_ref> [   
↪ <container_ref> ...]]]                                                [--insert-headers <header=value,...>] --   
↪ protocol-port <port>                                                 [--timeout-client-data <timeout>] [--   
↪ timeout-member-connect <timeout>]                                     [--timeout-member-data <timeout>] [--   
↪ timeout-tcp-inspect <timeout>]                                         [--enable | --disable] [--client-ca-tls-   
↪ container-ref <container_ref>]                                         [--client-authentication {NONE,OPTIONAL,   
↪ MANDATORY}]                                                         [--client-crl-container-ref <client_crl_   
↪ container_ref>]                                                       [--allowed-cidr [<allowed_cidr>]] [--wait]   
↪ [--tls-ciphers <tls_ciphers>]                                         [--tls-version [<tls_versions>]]   
                                <loadbalancer>
```

Create a listener

positional arguments:

<loadbalancer>            Load balancer for the listener (name or ID).

optional arguments:

-h, --help                show this help message and exit  
 --name <name>            Set the listener name.  
 --description <description>            Set the description of this listener.  
 --protocol {TCP,HTTP,HTTPS,TERMINATED\_HTTPS,UDP}            The protocol for the listener.  
 --connection-limit <limit>

(continues on next page)

(continued from previous page)

```

        Set the maximum number of connections permitted for this.
↪ listener.
    --default-pool <pool>
        Set the name or ID of the pool used by the listener if no L7.
↪ policies match.
    --default-tls-container-ref <container_ref>
        The URI to the key manager service secrets container containing.
↪ the certificate and key for TERMINATED_TLS
        listeners.
    --sni-container-refs [<container_ref> [<container_ref> ...]]
        A list of URIs to the key manager service secrets containers.
↪ containing the certificates and keys for
        TERMINATED_TLS the listener using Server Name Indication.
    --insert-headers <header=value,...>
        A dictionary of optional headers to insert into the request.
↪ before it is sent to the backend member.
    --protocol-port <port>
        Set the protocol port number for the listener.
    --timeout-client-data <timeout>
        Frontend client inactivity timeout in milliseconds. Default:
↪ 50000.
    --timeout-member-connect <timeout>
        Backend member connection timeout in milliseconds. Default: 5000.
    --timeout-member-data <timeout>
        Backend member inactivity timeout in milliseconds. Default:
↪ 50000.
    --timeout-tcp-inspect <timeout>
        Time, in milliseconds, to wait for additional TCP packets for.
↪ content inspection. Default: 0.
    --enable
        Enable listener (default).
    --disable
        Disable listener.
    --client-ca-tls-container-ref <container_ref>
        The URI to the key manager service secrets container containing.
↪ the CA certificate for TERMINATED_TLS
        listeners.
    --client-authentication {NONE,OPTIONAL,MANDATORY}
        The TLS client authentication verify options for TERMINATED_TLS.
↪ listeners.
    --client-crl-container-ref <client_crl_container_ref>
        The URI to the key manager service secrets container containing.
↪ the CA revocation list file for
        TERMINATED_TLS listeners.
    --allowed-cidr [<allowed_cidr>]
        CIDR to allow access to the listener (can be set multiple times).
    --wait
        Wait for action to complete
    --tls-ciphers <tls_ciphers>
        Set the TLS ciphers to be used by the listener in OpenSSL format.
    --tls-version [<tls_versions>]
        Set the TLS protocol version to be used by the listener (can be.
↪ set multiple times).
```

Below is an example of creating a listener for the load balancer `test-lb` which runs a HTTP protocol on port 80. This means any connections to the load balancer have to be done via this port as defined by the listener.

```
openstack loadbalancer listener create --name test-listener --protocol HTTP --protocol-
↪port 80 test-lb
```

*#This should return a table with details of the listener*

Field	Value
admin_state_up	True
connection_limit	-1
created_at	2020-11-16T09:10:22
default_pool_id	None
default_tls_container_ref	None
description	
id	ec01a05f-54be-419f-84cf-e89bba7e6acc
insert_headers	None
l7policies	
loadbalancers	fe22e256-d409-4f91-867f-508d08566892
name	test-listener
operating_status	OFFLINE
project_id	PROJECT_ID
protocol	HTTP
protocol_port	80
provisioning_status	PENDING_CREATE
sni_container_refs	[]
timeout_client_data	50000
timeout_member_connect	5000
timeout_member_data	50000
timeout_tcp_inspect	0
updated_at	None
client_ca_tls_container_ref	None
client_authentication	NONE
client_crl_container_ref	None
allowed_cidrs	None
tls_ciphers	
tls_versions	

> Listeners can refer to several pools.

## Pool and Members

A pool consists of members that serve traffic behind a load balancer, each member has a specified IP address and port to serve traffic.

**The following load balancer algorithms are supported:**

- **ROUND\_ROBIN:** each member of the pool is used in turn to handle incoming traffic.
- **LEAST\_CONNECTIONS:** a VM with the least number of connections is selected to receive the next incoming connection.
- **SOURCE\_IP and SOURCE\_IP\_PORT:** Source IP is hashed and divided by the total weight of the active VMs to determine which VM will receive the request.

**NOTE:** Pools can only be associated with one listener. **NOTE:** The recommended and preferred algorithm is SOURCE\_IP

Pools are created using `openstack loadbalancer pool create`:

```
openstack loadbalancer pool create [-h] [-f {json,shell,table,value,yaml}] [-c COLUMN] [-
↳noindent] [--prefix PREFIX]
                                [--max-width <integer>] [--fit-width] [--print-
↳empty] [--name <name>]
                                [--description <description>] --protocol {TCP,
↳HTTP,HTTPS,TERMINATED_HTTPS,PROXY,UDP}
                                (--listener <listener> | --loadbalancer <load_
↳balancer>)
                                [--session-persistence <session persistence>] -
↳-lb-algorithm
                                {SOURCE_IP,ROUND_ROBIN,LEAST_CONNECTIONS,
↳SOURCE_IP_PORT} [--enable | --disable]
                                [--tls-container-ref <container-ref>] [--ca-
↳tls-container-ref <ca_tls_container_ref>]
                                [--crl-container-ref <crl_container_ref>] [--
↳enable-tls | --disable-tls] [--wait]
                                [--tls-ciphers <tls_ciphers>] [--tls-version [
↳<tls_versions>]]
```

optional arguments:

```
-h, --help                show this help message and exit
--name <name>             Set pool name.
--description <description>
                           Set pool description.
--protocol {TCP,HTTP,HTTPS,TERMINATED_HTTPS,PROXY,UDP}
                           Set the pool protocol.
--listener <listener>
                           Listener to add the pool to (name or ID).
--loadbalancer <load_balancer>
                           Load balancer to add the pool to (name or ID)
--session-persistence <session persistence>
                           Set the session persistence for the listener (key=value).
--lb-algorithm {SOURCE_IP,ROUND_ROBIN,LEAST_CONNECTIONS,SOURCE_IP_PORT}
                           Load balancing algorithm to use.
--enable                  Enable pool (default).
--disable                 Disable pool.
--tls-container-ref <container-ref>
                           The reference to the key manager service secrets container
↳containing the certificate and key for
                           ``tls_enabled`` pools to re-encrypt the traffic to backend member
↳servers.
--ca-tls-container-ref <ca_tls_container_ref>
                           The reference to the key manager service secrets container
↳containing the CA certificate for
                           ``tls_enabled`` pools to check the backend member servers
↳certificates
--crl-container-ref <crl_container_ref>
                           The reference to the key manager service secrets container
↳containing the CA revocation list file for
```

(continues on next page)

(continued from previous page)

```

        ``tls_enabled`` pools to validate the backend member servers.
↪certificates.
    --enable-tls          Enable backend member re-encryption.
    --disable-tls        Disable backend member re-encryption.
    --wait               Wait for action to complete
    --tls-ciphers <tls_ciphers>
                        Set the TLS ciphers to be used by the pool in OpenSSL cipher.
↪string format.
    --tls-version [<tls_versions>]
                        Set the TLS protocol version to be used by the pool (can be set.
↪multiple times).

```

Then members can be defined for the pool using *openstack loadbalancer member create*:

```

openstack loadbalancer member create [-h] [-f {json,shell,table,value,yaml}] [-c COLUMN]
↪[--noindent] [--prefix PREFIX]
                                [--max-width <integer>] [--fit-width] [--
↪print-empty] [--name <name>]
                                [--disable-backup | --enable-backup] [--
↪weight <weight>] --address <ip_address>
                                [--subnet-id <subnet_id>] --protocol-port
↪<protocol_port>
                                [--monitor-port <monitor_port>] [--monitor-
↪address <monitor_address>]
                                [--enable | --disable] [--wait]
                                <pool>

```

Creating a member in a pool

positional arguments:

<pool> ID or name of the pool to create the member for.

optional arguments:

```

-h, --help                show this help message and exit
--name <name>             Name of the member.
--disable-backup          Disable member backup (default)
--enable-backup           Enable member backup
--weight <weight>         The weight of a member determines the portion of requests or
↪connections it services compared to the other
                        members of the pool.
--address <ip_address>    The IP address of the backend member server
--subnet-id <subnet_id>   The subnet ID the member service is accessible from.
--protocol-port <protocol_port>
                        The protocol port number the backend member server is listening
↪on.
--monitor-port <monitor_port>
                        An alternate protocol port used for health monitoring a backend
↪member.
--monitor-address <monitor_address>
                        An alternate IP address used for health monitoring a backend

```

(continues on next page)

(continued from previous page)

```
↪ member.
  --enable          Enable member (default)
  --disable         Disable member
  --wait            Wait for action to complete
```

## Example

```
#Use openstack loadbalancer pool create to define the pool which is the default pool for ↪
↪ test-listener
openstack loadbalancer pool create --name test-pool --lb-algorithm ROUND_ROBIN --
↪ listener ec01a05f-54be-419f-84cf-e89bba7e6acc --protocol HTTP

#This should return a table with details of the pool
```

Field	Value
admin_state_up	True
created_at	2020-11-16T09:16:31
description	
healthmonitor_id	
id	733b78a8-4e0d-4d31-9eb4-e8084a94dac0
lb_algorithm	ROUND_ROBIN
listeners	ec01a05f-54be-419f-84cf-e89bba7e6acc
loadbalancers	fe22e256-d409-4f91-867f-508d08566892
members	
name	test-pool
operating_status	OFFLINE
project_id	PROJECT_ID
protocol	HTTP
provisioning_status	PENDING_CREATE
session_persistence	None
updated_at	None
tls_container_ref	None
ca_tls_container_ref	None
crl_container_ref	None
tls_enabled	False
tls_ciphers	
tls_versions	

```
#Then we can define the pool members using the command 'openstack loadbalancer member ↪
↪ create'
openstack loadbalancer member create --subnet-id <private-subnet-id> --address <server-
↪ ip> --protocol_port 80 test-pool
```

After setting up the load balancer, listener, and pool we can test that we can access the VM in the pool via the load balancer:

```
ssh <user>@<lb-floating-IP> -p <listener-port>
```

More load balancing examples can be found here: <https://docs.openstack.org/octavia/train/user/guides/basic-cookbook.html>

## Load Balancer Status

We can also check the status of a load balancing pool from the command line. For example, we can check the status of a load balancer that is in front of a kubernetes service being hosted on multiple nodes.

To check the load balancer status we can use the following command:

```
openstack loadbalancer status show <loadbalancer-id>
```

*#This will give a load balancer status in the form similar to the following:*

```
{
  "loadbalancer": {
    "listeners": [
      {
        "pools": [
          {
            "name": "kubernetes-pool-t3ue6p7zvss5",
            "health_monitor": {
              "provisioning_status": "ACTIVE",
              "type": "TCP",
              "id": "6137160c-7835-48f8-b79a-214359f48a92",
              "operating_status": "ONLINE",
              "name": ""
            },
            "provisioning_status": "ACTIVE",
            "members": [
              {
                "name": "cluster-node-0",
                "provisioning_status": "ACTIVE",
                "address": "10.0.0.131",
                "protocol_port": 30858,
                "id": "edb62670-8b79-4912-b449-6269f0ff3076",
                "operating_status": "OFFLINE"
              },
              {
                "name": "cluster-node-1",
                "provisioning_status": "ACTIVE",
                "address": "10.0.0.8",
                "protocol_port": 30858,
                "id": "564e4f40-9e2c-4eb2-877b-ce4169b4faa9",
                "operating_status": "OFFLINE"
              }
            ],
            "id": "1cef96d9-d429-413b-a92f-daf2bc0d419d",
            "operating_status": "OFFLINE"
          }
        ],
        "provisioning_status": "ACTIVE",
        "id": "4ff15be3-71f8-475e-81ea-5dfcb1b92e0f",
```

(continues on next page)

(continued from previous page)

```

        "operating_status": "OFFLINE",
        "name": "kubernetes-listener-tnroqym5ruw"
    },
    ],
    "provisioning_status": "ACTIVE",
    "id": "56c1563f-9afc-4deb-9ba4-ea99e9b71547",
    "operating_status": "OFFLINE",
    "name": "kubernetes-lb-yvawgzx7acca"
}
}

```

## Health Monitor

It is possible to set up a listener without a health monitor, however OpenStack recommends to always configure production load balancers with a health monitor. Health monitors are attached to load balancer pools and monitor the health of each pool member. Should a pool member fail a health check, the health monitor should remove that member temporarily from the pool. If that pool member begins to respond to health checks, the health monitor returns the member to the pool.

### Health Monitor Options

- **delay:** number of seconds to check between health checks
- **timeout:** the number of seconds to any given health check to complete. Timeout should always be smaller than delay.
- **-max-retries:** number of subsequent health checks a given back-end server must fail before it is considered down, or that a failed back-end server must pass to be considered up again.

### HTTP Health Monitors

By default, Octavia will probe the “/” path on the application server. However, in many applications this is not appropriate because the “/” path ends up being a cached page, or causes the server to do more work than it is necessary for a basic health check.

HTTP health monitors also have the following options:

- **url\_path:** path part of the URL that should be retrieved from the back end server. By default it is “/”.
- **http\_method:** HTTP method that should be used to retrieve the url\_path. By default this is “GET”.
- **expected\_codes:** list of HTTP status codes that indicate an OK health check. By default this is just “200”.

To create health monitors and attach them to load balancing pools, we can use the command:

```

openstack loadbalancer healthmonitor create [-h] [-f {json,shell,table,value,yaml}] [-c,
↪ COLUMN] [--noindent]
                                     [--prefix PREFIX] [--max-width
↪ <integer>] [--fit-width] [--print-empty]
                                     [--name <name>] --delay <delay> [--
↪ domain-name <domain_name>]
                                     [--expected-codes <codes>]
                                     [--http-method {GET,POST,DELETE,PUT,
↪ HEAD,OPTIONS,PATCH,CONNECT,TRACE}]
                                     [--http-version <http_version>] --
↪ timeout <timeout> --max-retries <max_retries>

```

(continues on next page)

(continued from previous page)

```

[--url-path <url_path>] --type {PING,
↪ HTTP, TCP, HTTPS, TLS-HELLO, UDP-CONNECT}
[--max-retries-down <max_retries_down>
↪ ] [--enable | --disable] [--wait]
<pool>

#positional argument
<pool> define pool for the health monitor

#optional arguments:
--name <name>           Set the health monitor name.
--delay <delay>         Set the time in seconds, between sending probes to members.
--domain-name <domain_name>
                        Set the domain name, which be injected into the HTTP Host Header.
↪ to the backend server for HTTP health
                        check.
--expected-codes <codes>
                        Set the list of HTTP status codes expected in response from the
↪ member to declare it healthy.
--http-method {GET,POST,DELETE,PUT,HEAD,OPTIONS,PATCH,CONNECT,TRACE}
                        Set the HTTP method that the health monitor uses for requests.
--http-version <http_version>
                        Set the HTTP version.
--timeout <timeout>     Set the maximum time, in seconds, that a monitor waits to connect.
↪ before it times out. This value must be
                        less than the delay value.
--max-retries <max_retries>
                        The number of successful checks before changing the operating
↪ status of the member to ONLINE.
--url-path <url_path>
                        Set the HTTP URL path of the request sent by the monitor to test.
↪ the health of a backend member.
--type {PING,HTTP,TCP,HTTPS,TLS-HELLO,UDP-CONNECT}
                        Set the health monitor type.
--max-retries-down <max_retries_down>
                        Set the number of allowed check failures before changing the
↪ operating status of the member to ERROR.
--enable                Enable health monitor (default).
--disable               Disable health monitor.
--wait                 Wait for action to complete

```

## 16.1.4 Status Codes

The load balancer, listener, health monitor, pool and pool member have a provisioning status and an operating status.

### Provisioning Status Codes

Code	Reason
<b>ACTIVE</b>	Entity Provisioned successfully
<b>DELETED</b>	Entity successfully deleted
<b>ERROR</b> Please refer to error messages	Provisioning failed
<b>PENDING_CREATE</b>	Entity is being created
<b>PENDING_UPDATE</b>	Entity is being updated
<b>PENDING_DELETE</b>	Entity is being deleted

### Operation Status Codes

Status	Reason
<b>ONLINE</b>	Entity is operating normally. All Pool members are healthy
<b>DRAINING</b>	Member is not accepting new connections
<b>OFFLINE</b>	Entity is administratively disabled
<b>DEGRADED</b>	One or more components are in ERROR
<b>ERROR</b> Please refer to error messages	Entity has failed. Member is failing health monitoring check. All pool members are in ERROR
<b>NO_MONITOR</b>	No health monitor is configured. Current status is known

## 16.1.5 References

<https://docs.openstack.org/octavia/train/reference/introduction.html>

<https://docs.openstack.org/octavia/train/user/guides/basic-cookbook.html>

<https://docs.openstack.org/octavia/train/reference/glossary.html>

<https://wiki.openstack.org/wiki/Neutron/LBaaS/Deprecation>

<http://cbonte.github.io/haproxy-dconv/1.8/configuration.html#4-balance>

<https://docs.openstack.org/api-ref/load-balancer/v2/index.html?expanded=create-pool-detail,show-pool-details-detail#general-api-overview>

<https://docs.openstack.org/mitaka/networking-guide/config-lbaas.html>

## 16.2 Load Balancer as a Service Quick Start

This quick start assumes you have a VM set up with SSH and a webserver that you want to expose. The steps included are applicable to other ports and services too. This example assumes you've configured your machine following [Running an NGinX webserver inside a Docker container](#). The steps will be similar for bespoke systems too.

First we will create a load balancer and use it to provide ssh access to a VM initially, then add another listener and pool to point to the webserver on the VM.

### 16.2.1 Create a Loadbalancer

In the Openstack web interface, expand Network and then click Load Balancers on the left hand side of the screen.



## Project

API Access

Compute

Rating

Volumes

Container Infra

Network

Network Topology

Networks

Routers

Security Groups

**Load Balancers**

Floating IPs

Orchestration

## Identity

Click “Create Load Balancer”



On the “Load Balancer Details” screen enter a name for the load balancer and select the subnet for your projects private network. Leave the IP Address and Flavor fields blank. Then click “Next”

Create Load Balancer

×

?

Load Balancer Details

Listener Details \*

Pool Details \*

Pool Members

Monitor Details \*

Provide the details for the load balancer.

Name

Demo

IP address

Description

Flavour

Subnet \*

perfsonar-testbed: 192.168.248.0/24 (perfsonar-testbed)

Admin State Up

YES

NO

×

CANCEL

←

BACK

NEXT

→

CREATE LOAD BALANCER

On the “Listener Details” screen select Protocol TCP and set port 200 (which we’ve randomly selected). To avoid conflicts it’s best to select random port numbers greater than 1024. Enter the name as ssh. The other defaults are fine. Click “Next”

## Create Load Balancer

Load Balancer Details

Listener Details

Pool Details \*

Pool Members

Monitor Details \*

Provide the details for the listener.

Name

ssh

Description

Protocol \*

TCP

Port \*

200

Client Data Timeout

50000

TCP Inspect Timeout

0

Member Connect Timeout

5000

Member Data Timeout

50000

Connection Limit \*

-1

Admin State Up

YES

NO

× CANCEL

← BACK

NEXT →

CREATE LOAD BALANCER

On the “Pool Details” screen enter the name as SSH and set the Algorithm and Session Persistence to “Source IP”. Click “Next”

## Create Load Balancer

×

?

Load Balancer Details

Listener Details

**Pool Details**

Pool Members

Monitor Details \*

Provide the details for the pool.

Name

ssh

Description

Algorithm \*

SOURCE\_IP

Session Persistence

SOURCE\_IP

Admin State Up

YES

NO

×

CANCEL

←

BACK

NEXT

→

CREATE LOAD BALANCER

On the “Pool Members” screen add the VM you want to point at and set the port to 22. Click “Next”

Create Load Balancer

×

?

Load Balancer Details

Listener Details

Pool Details

Pool Members

Monitor Details \*

Add members to the load balancer pool.

Allocated Members 1

IP Address *	Subnet *	Port *	Weight	
> 172.16.114.212	Internal4	20	1	REMOVE

ADD EXTERNAL MEMBER

Available Instances

Filter

Name	IP Address	
test cleanup repos	172.16.113.42	ADD
demo private	192.168.248.15	ADD
demo 1	172.16.114.212	ADD
Test gnocchi grafana	172.16.103.105	ADD

×

 CANCEL

← BACK

NEXT →

CREATE LOAD BALANCER

On the “Monitor Details” screen enter the name ssh and set the type to “TCP”. Click “Create Load Balancer”

Create Load Balancer

×

?

Load Balancer Details

Listener Details

Pool Details

Pool Members

Monitor Details

Provide the details for the health monitor.

Name

ssh

Type \*

TCP

Max Retries Down \*

3

Delay (sec) \*

5

Max Retries \*

3

Timeout (sec) \*

5

Admin State Up

YES

NO

×

 CANCEL

← BACK

NEXT →

CREATE LOAD BALANCER

You will then see the loadbalancer that you created in the list with the Provisioning Status of “Pending Create”

## Load Balancers

Click here for filters or full text search. × + CREATE LOAD BALANCER DELETED LOAD BALANCERS

Displaying 1 item

<input type="checkbox"/>	Name ^	IP Address	Operating Status	Provisioning Status	Admin State Up	
<input type="checkbox"/>	> Demo	192.168.248.185	Offline	Pending Create	Yes	EDIT LOAD BALANCER <span>▾</span>

Displaying 1 item

© Copyright STFC 2016 | [Terms of Service](#) | [Service-level Agreement](#) | [Frequently Asked Questions](#) | [GitHub](#) | [Issue Tracker](#)

After a few minutes the Provisioning Status will be updated to “Active”

## Load Balancers

Click here for filters or full text search. × + CREATE LOAD BALANCER DELETED LOAD BALANCERS

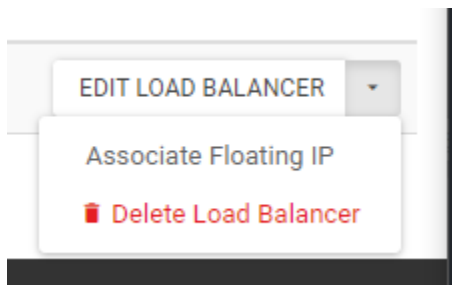
Displaying 1 item

<input type="checkbox"/>	Name ^	IP Address	Operating Status	Provisioning Status	Admin State Up	
<input type="checkbox"/>	> Demo	192.168.248.185	Offline	Active	Yes	EDIT LOAD BALANCER <span>▾</span>

Displaying 1 item

© Copyright STFC 2016 | [Terms of Service](#) | [Service-level Agreement](#) | [Frequently Asked Questions](#) | [GitHub](#) | [Issue Tracker](#)

Now Click the arrow next to “Edit Load Balancer” and click “Associate Floating IP”



Select one of your floating IPs and then click “Associate”

# Associate Floating IP Address

Select a floating IP address to associate with the load balancer or a floating IP pool in which to allocate a new floating IP address.

Floating IP address or pool \*

130.246.215.87 ▼

× CANCEL

✓ ASSOCIATE

You should now be able to ssh to your VM on port 200 of the IP you assigned to the load Balancer

```
ssh username@130.246.215.87 -p 200
```

## 16.2.2 Viewing details

If you click on the Name of the load balancer you can view details of the listeners and pools

Unfortunately a number of the “Operating Status” fields always report “Offline” even when they are online

You can add another Listener to this load balancer by hitting “Create Listener” and following similar steps to the above  
:ref: *create\_octavia\_listener*

Try using these steps to load balance access to the nginx webserver created in the other tutorial.

## 16.2.3 Explanation of terms

Algorithms:

Least connections - sends each connection to whichever pool member has the

Round Robin - sends connections to each backend node in return

Source IP - Sends connections to backend nodes based on the hash of the client’s IP

Session Persistence:

Source IP - Uses the hash of the client’s IP to maintain session persistence

HTTP Cookie - Session persistence based on http cookies

APP Cookie - Session persistence based on application cookies

Monitor Types:

HTTP - Checks for a configurable http response

HTTPS - Checks for a http 200. Requires that the certificate is trusted by the loadbalancer and we currently dont have a way to inject new CAs

PING - A simple ping test

TCP - A simple tcp handshake

TLS-HELLO - A tls handshake

UDP-CONNECT - Checks for an open udp socket.

## 16.3 LBaaS v2: Heat Stacks

Load balancing in a template consists of:

- **Pool:** A group of servers which are identified by their floating IP addresses. The size of the pool can be increased or decreased in response to traffic to the instances detected by the load balancer.
- **Listener:** Represent a listening endpoint for the vip. Listener resources listen for the client traffic.
- **Health Monitor:** Monitors the health of the pool.
- **Load Balancer:** Balances the traffic to server pools. In combination with the health monitor and listener, the load balancer can redistribute traffic to servers in the event of one server failing for example.

### 16.3.1 Octavia Resources in Heat

```
pool:
  type: OS::Octavia::Pool
  properties:
    ## required #####
    lb_algorithm: String #algorithm to distribute the load between the members of the
    ↪pool: ROUND_ROBIN, LEAST_CONNECTIONS, SOURCE_IP
    protocol: String #protocol of the pool: TCP, HTTP, HTTPS, TERMINATED_HTTPS, SOURCE_IP

    ## optional #####
    admin_state_up: Boolean #the administrative state of the pool
    description: String #description of the pool
    listener: String #listener name/ID to be associated with the pool
    loadbalancer: String #the loadbalancer name/ID to be associated to the pool
    name: String #name of this pool
    session_persistence: {"type": String, "cookie_name": String} #configuration of
    ↪session persistence.
    #required - type: the method of session persistence feature: SOURCE_IP, HTTP_
    ↪COOKIE, APP_COOKIE
    #optional - cookie_name: name of the cookie - required if type is APP_COOKIE
    tls_enabled: Boolean #default - false. Enable backend member re-encryption

listener:
  type: OS::Octavia::Listener
  properties:
    ## required #####
    protocol: String # protocol on which to listen for client traffic: TCP, HTTP, HTTP/S,
    ↪TERMINATED_HTTPS, PROXY, UDP
    protocol_port: Integer # TCP or UDP Port on which to listen for client traffic

    ## optional #####
    admin_state_up: Boolean
```

(continues on next page)

(continued from previous page)

```

allowed_cidrs: [String, String, ...]
connection_limit: Integer
default_pool: String
default_tls_container_ref: String
description: String
loadbalancer: String
name: String
sni_container_refs: [Value, Value, ...]
tenant_id: String

health_monitor:
  type: OS::Octavia::HealthMonitor
  properties:
    ## required properties #####
    delay: Integer #(seconds) minimum time between regular connections of pool member
    max_retries: Integer # max number of permissible connection failures before changing_
    ↪ member status to INACTIVE
    pool: String #name/ID of load balancing pool (type: octavia.pool)
    timeout: Integer # maximum number of seconds for a monitor to wait for a connection_
    ↪ to be established before timeout.
    type: String #type of health monitor: PING, TCP, HTTP, HTTPS, UDP-CONNECT
    ## Optional #####
    admin_state_up: Boolean #(default: true) administrative state of a health monitor.
    expected_codes: String #expected HTTP codes for a healthy monitor e.g. a single_
    ↪ value (200), a list (200,202), or a range (202-204)
    http_method: String #method used for requests by HTTP monitor
    tenant_id: String #ID of tenant which owns the monitor
    url_path: String #HTTP path used in the HTTP request used by the monitor to test a_
    ↪ member's health

load_balancer:
  type: OS::Octavia::LoadBalancer
  properties:
    ## required #####
    vip_subnet: String #name/ID of the subnet on which to allocate the VIP address
    ## optional #####
    admin_state_up: Boolean #administrative state of the load balancer (default: true)
    description: String #description of the load balancer
    flavor: String #name/id of the flavor of the load balancer
    name: String #name of the loadbalancer
    provider: String #the provider of this load balancer
    tenant_id: String #id of the tenant which owns the loadbalancer
    vip_address: String #IP address for the VIP

```

The table below lists the attributes for each resource:

Resource	Attributes
↪	
↪	
↪	
↪	

(continues on next page)

(continued from previous page)

```

| OS::Octavia::LoadBalancer      | **flavor_id:** The flavor ID of the LoadBalancer
↳<br><br>**pools:** the pools this LoadBalancer is associated with<br><br>**show:**
↳detailed information about the resource<br><br>**vip_address:** the VIP
↳addresses of the LoadBalancer<br><br>**vip_port_id:** the VIP port of the LoadBalancer
↳<br><br>**vip_subnet_id:** the VIP subnet of LoadBalancer |
| OS::Octavia::Pool              | **healthmonitor_id:** the ID of the health monitor
↳associated with this pool<br><br>**listeners:** listener associated with this pool <br>
↳<br>**members:** members associated with this pool<br><br>**show:** detailed
↳information about resource
↳
↳
| OS::Octavia::HealthMonitor     | **pool:** the list of pools related to this monitor
↳<br><br>**show:** detailed information about the resource
↳
↳
↳
| OS::Octavia::Listener          | **default_pool_id:** ID of the default pool the
↳listener is associated to<br><br>**loadbalancers:** the ID of the load balancer this
↳listener is associated to<br><br>**show:** detailed information about resource
↳
↳
↳

```

## Example

The example below shows how the Octavia resources can be defined in a Heat Template.

```

#HTTP health monitor
health_monitor:
  type: OS::Octavia::HealthMonitor
  properties:
    delay: 3 #three second delay
    type: "HTTP"
    timeout: 3 # seconds
    max_retries: 3
    pool: {get_resource: pool}
    url_path: /healthcheck #this is a URL path that is configured on the servers in the
↳pool that the monitor can reach to check pool health.

pool:
  type: OS::Octavia::Pool
  properties:
    lb_algorithm: "LEAST_CONNECTIONS" #the preferred algorithm
    protocol: "HTTP"
    listener: {get_resource: listener}

member:
  type: OS::Octavia::PoolMember
  properties:
    address: {get_attr: [server, first_address]}
    pool: {get_resource: pool}
    protocol_port: 80
    subnet: {get_param: private_subnet}

```

(continues on next page)

(continued from previous page)

```
listener:
  type: OS::Octavia::Listener
  properties:
    protocol: "HTTP"
    protocol_port: 80 # listen on the HTTP port
    loadbalancer: {get_resource: lb}

lb:
  #define the load balancer and the private subnet to use
  type: OS::Octavia::LoadBalancer
  properties:
    vip_subnet: <private-subnet-id>

# Attach a floating IP to the load balancer
floating_ip_association:
  # Associate a floating IP to the Load Balancer so that it can be accessed
  # using an external IP
  type: OS::Neutron::FloatingIPAssociation
  properties:
    floatingip_id: <floating-ip-id>
    port_id: {get_attr: [lb,vip_port_id]}
```

### 16.3.2 References

<https://ibm-blue-box-help.github.io/help-documentation/heat/autoscaling-with-heat/>

[https://docs.openstack.org/heat/latest/template\\_guide/openstack.html](https://docs.openstack.org/heat/latest/template_guide/openstack.html)

<https://docs.openstack.org/octavia/train/reference/introduction.html>

<https://docs.openstack.org/octavia/train/user/guides/basic-cookbook.html>

## 17.1 Introducing Swift

Swift is a highly available, distributed, eventually consistent object/blob store. Due to the flat structure of object storage, it is highly scalable, allowing large amounts of unstructured data to be stored efficiently.

Files and data are uploaded as **objects**, which are organised into **containers**. Once created, object data cannot be modified without uploading the entire object again, making this storage unsuitable for frequently updated data.

---

**Note:** Swift containers should not be confused with Docker containers. Containers may also be referred to as buckets or pools.

---

Although you cannot nest directories in object storage, a hierarchical structure can be simulated within a container through the inclusion of delimiters ('/') in object names.

### 17.1.1 Explanation of Terms

**Object Store:** This provides a system for data storage that enables users to access the same data, both as an object and as a file, simplifying management and controlling storage costs.

**Account:** The top level of the Object Storage system hierarchy, defining a namespace for containers. Users own all resources in their account. Synonymous with a *project* or *tenant*.

**Container:** This defines a namespace for objects, providing a storage compartment for and a way to organise data, similar to directories in a Linux system. However, unlike directories, containers cannot be nested. Data must be stored in a container, and so objects are created within containers. Containers may also be referred to as buckets or pools.

**Object:** The basic storage entity and any optional metadata that represents the data being stored. When data is uploaded, the data is stored as-is, with no compression or encryption.

**Folder/Pseudo-folder:** Similar to folders in a desktop operating system. They are virtual collections defined by a common prefix in objects' names. Placeholder objects may also exist to represent folders.

**Eventually Consistent:** When data is updated, it is guaranteed that this will be reflected across all nodes eventually. This allows high availability, but there is no guarantee for when this will happen, so changes to data may not be reflected in read requests immediately.

**Unstructured Data:** Data that does not conform to, or cannot be easily organised into, a traditional relational database, including emails, videos and web pages.

**Access Control List (ACL):** Used to control access to objects within a container. An ACL cannot be stored with individual objects.

## 17.1.2 References

<https://www.openstack.org/software/releases/train/components/swift>

<https://docs.openstack.org/swift/train/>

<https://www.redhat.com/en/topics/data-storage/file-block-object-storage>

[https://access.redhat.com/documentation/en-us/red\\_hat\\_gluster\\_storage/3.1/html/administration\\_guide/chap-managing\\_object\\_store](https://access.redhat.com/documentation/en-us/red_hat_gluster_storage/3.1/html/administration_guide/chap-managing_object_store)

[https://access.redhat.com/documentation/en-us/red\\_hat\\_gluster\\_storage/3.1/html/administration\\_guide/components\\_of\\_object\\_storage](https://access.redhat.com/documentation/en-us/red_hat_gluster_storage/3.1/html/administration_guide/components_of_object_storage)

[https://docs.openstack.org/swift/train/api/object\\_api\\_v1\\_overview.html](https://docs.openstack.org/swift/train/api/object_api_v1_overview.html)

<https://docs.openstack.org/swift/train/api/pseudo-hierarchical-folders-directories.html>

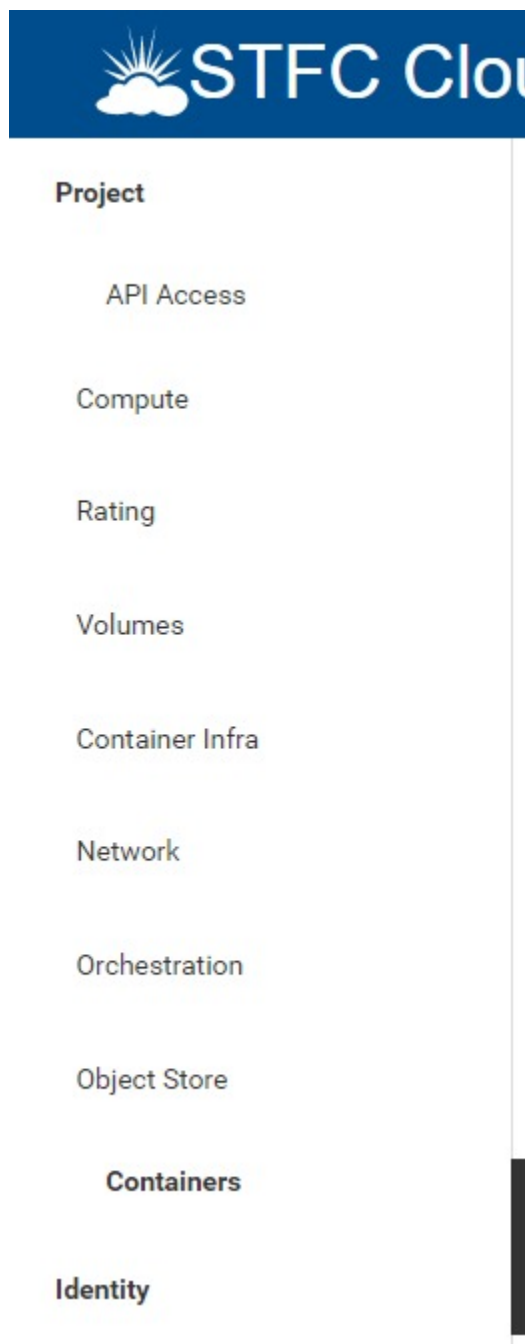
[https://docs.openstack.org/swift/train/overview\\_acl.html](https://docs.openstack.org/swift/train/overview_acl.html)

<https://www.ibm.com/cloud/learn/object-storage>

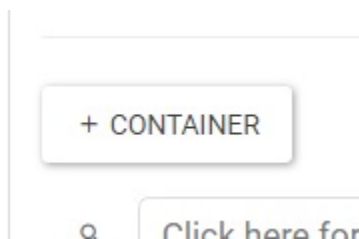
## 17.2 Object Storage Quick Start

### 17.2.1 Create a Container

In the Openstack web interface, expand *Object Store* and then click *Containers* on the left hand side of the screen.



Click **+** *Container*.



On the *Create Container* screen, enter a name for the container and select whether the container should be publicly accessible, then click *Submit*.

## Create Container



Container Name \*



CONTAINER\_1



Container name must not contain "/".

Container Access

PUBLIC

NOT PUBLIC

A Public Container will allow anyone with the Public URL to gain access to your objects in the container.

× CANCEL

✓ SUBMIT

**Warning:** Selecting “public” will allow the contents of a container to be viewed by anyone, with no authentication required. See [Editing Container Metadata](#) or [Containers](#) for greater control over read and write access.

If successful, the container will be listed below the + *Container* button.

+ CONTAINER



Click here for filters or full text search. ×

CONTAINER\_1

Details of the container may be viewed by clicking on the listing. This will also display a checkbox that may be clicked to enable or disable public access.

CONTAINER\_1



Object Count:	0
Size:	0 bytes
Date Created:	Apr 25, 2022
<input type="checkbox"/> Public Access:	Disabled

## 17.2.2 Upload a File

Having selected a listed container, click the *Upload* icon next to + *Folder*.



Select a file to upload, and a unique name for it in the container. Then click *Upload File*.

# Upload File To: CONTAINER\_1

×

File \*

Choose File

FILE\_1.txt

File Name

FILE\_1.txt

Note: Delimiters ('/') are allowed in the file name to place the new file into a folder that will be created when the file is uploaded (to any depth of folders).

×

 CANCEL
 

↑
 UPLOAD FILE

**Warning:** Files uploaded in this way must be less than 5GiB. See [Large Files](#) for instructions on uploading larger files. If *Error: Unable to upload the object* is displayed for a file smaller than this, you may need to contact the Cloud team to request a quota increase.

If successful, the file will be listed below the *Upload* icon.

### CONTAINER\_1

×

↑

+ FOLDER

Click here for filters or full text search.

×

Displaying 1 item

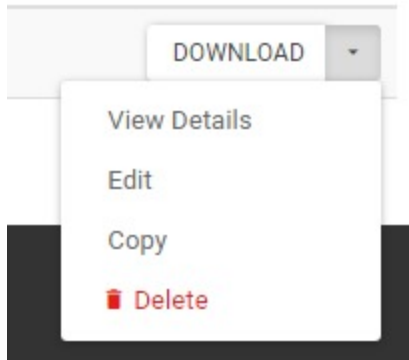
<input type="checkbox"/>	Name ^	Size	
<input type="checkbox"/>	FILE_1.txt	4 bytes	<div> <div>DOWNLOAD</div> <div>▼</div> </div>

Displaying 1 item

To view file details, select the arrow next to *Download* to view the options, then click *View Details*.

17.2. Object Storage Quick Start

251



The file details will then be shown.

## Object Details



<b>Name</b>	FILE_1.txt
<b>Hash</b>	0cbc6611f5540bd0809a388dc95a615b
<b>Content Type</b>	text/plain
<b>Timestamp</b>	Apr 26, 2022 3:20:10 PM
<b>Size</b>	4 bytes

× CLOSE

### 17.2.3 Create a Folder

To create an empty folder, click + *Folder*.



Enter a unique folder name, then click *Create Folder*.

## Create Folder In: CONTAINER\_1



Folder Name

Note: Delimiters ('/') are allowed in the folder name to create deep folders.

× CANCEL

+ CREATE FOLDER

Alternatively, folders may be created by including a new folder name, followed by a delimiter ('/'), before the name of an uploaded file.

## Upload File To: CONTAINER\_1



File \*

FILE\_1.txt

Note: Delimiters ('/') are allowed in the file name to place the new file into a folder that will be created when the file is uploaded (to any depth of folders).

File Name

× CANCEL

⬆️ UPLOAD FILE

If successful, the folder will be listed with other files in the container.

## CONTAINER\_1

Displaying 3 items

<input type="checkbox"/> Name ^	Size	
<input type="checkbox"/> FILE_1.txt	4 bytes	<input type="button" value="DOWNLOAD"/> <input type="button" value="v"/>
<input type="checkbox"/> FOLDER_1	Folder	<input type="button" value="🗑 DELETE"/>
<input type="checkbox"/> FOLDER_2	Folder	<input type="button" value="🗑 DELETE"/>

Displaying 3 items

The contents of a folder may be viewed by clicking the folder listing.

## CONTAINER\_1 : FOLDER\_1

Displaying 1 item

<input type="checkbox"/> Name ^	Size	
<input type="checkbox"/> FILE_1.txt	4 bytes	<input type="button" value="DOWNLOAD"/> <input type="button" value="v"/>

Displaying 1 item

### 17.2.4 Create Deep Folders

When viewing a folder, a deep folder can be created by selecting + *Folder*, as *previously* when viewing a container.

## Create Folder In: CONTAINER\_1 : FOLDER\_1 ✕

Folder Name

Note: Delimiters ('/') are allowed in the folder name to create deep folders.

✕ CANCEL

+ CREATE FOLDER

Multiple levels of folders may be created by including delimiters ('/') between each folder name. Any folders specified that do not exist will be created.

## Create Folder In: CONTAINER\_1 : FOLDER\_1 ✕

Folder Name

Note: Delimiters ('/') are allowed in the folder name to create deep folders.

✕ CANCEL

+ CREATE FOLDER

This can equivalently be done when viewing the container, outside all folders.

## Create Folder In: CONTAINER\_1 ✕

Folder Name

Note: Delimiters ('/') are allowed in the folder name to create deep folders.

✕ CANCEL

+ CREATE FOLDER

Alternatively, the new folder names may be included when uploading a file.

# Upload File To: CONTAINER\_1



File \*

 FILE\_1.txt

Note: Delimiters ('/') are allowed in the file name to place the new file into a folder that will be created when the file is uploaded (to any depth of folders).

File Name

As when viewing a container, the next level of folders will be listed alongside files in the folder you are viewing. Deeper folders and files will not be displayed without navigating into their parent folders by selecting them.

## CONTAINER\_1 : FOLDER\_1



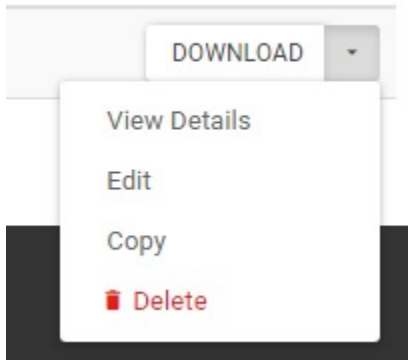
Displaying 3 items

<input type="checkbox"/>	Name ^	Size	
<input type="checkbox"/>	FILE_1.txt	4 bytes	<input type="button" value="DOWNLOAD"/> ▾
<input type="checkbox"/>	FOLDER_2	Folder	<input type="button" value="DELETE"/>
<input type="checkbox"/>	FOLDER_3	Folder	<input type="button" value="DELETE"/>

Displaying 3 items

### 17.2.5 Delete a File

To delete an individual file, select the arrow next to *Download* to view the file options, then clicking *Delete*.



On the *Delete Files* screen, select *Delete* to begin deletion.

## Delete Files in CONTAINER\_1



Collecting information for deletion: 1 files in 0 folders.

× CANCEL

DELETE

If successful, a progress bar should fill, and the *Delete Files* screen can be dismissed by clicking *OK* or *X*.

### 17.2.6 Delete a Folder

To delete a folder and its contents, click the *Delete* button.



As for a file, on the *Delete Files* screen, select *Delete* to begin deletion.

## Delete Files in CONTAINER\_1



Collecting information for deletion: 3 files in 1 folders.

× CANCEL

DELETE

If successful, a progress bar should fill, and the *Delete Files* screen can be dismissed by clicking *OK* or *X*.

### 17.2.7 Delete Multiple Files or Folders

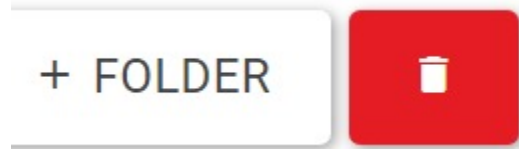
Select the files and/or folders to be deleted by clicking the checkbox next to each. All file and folder in the current folder or container can be selected by clicking the checkbox next to *Name*.

Displaying 4 items

<input type="checkbox"/>	Name ^	Size	
<input checked="" type="checkbox"/>	FILE_1.txt	4 bytes	DOWNLOAD ▾
<input type="checkbox"/>	FILE_2.txt	4 bytes	DOWNLOAD ▾
<input checked="" type="checkbox"/>	FOLDER_1	Folder	DELETE
<input type="checkbox"/>	FOLDER_2	Folder	DELETE

Displaying 4 items

Click the *Delete* icon next to + *Folder*.



As for individual files and folders, select *Delete* on the *Delete Files* screen to begin deletion. You can then dismiss the *Delete Files* screen.

### 17.2.8 Delete a Container

In order to delete a container, all files and folders in the container must first be deleted. Then, with the container selected, click the *Delete* icon next to its name.



On the the *Confirm Delete* screen, select *Delete* to begin deletion.

## Confirm Delete



Are you sure you want to delete container CONTAINER\_1?

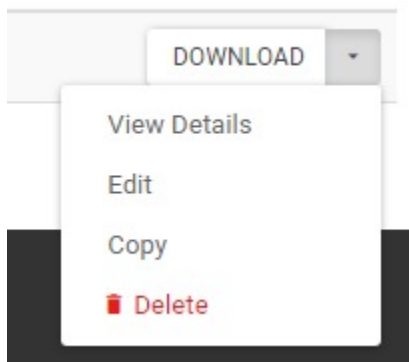
CANCEL

DELETE

If successful, a success message will appear, and the container will no longer be listed.

### 17.2.9 Edit a File

To edit a file, select the arrow next to *Download* to view the options, then click *Edit*.



On the *Edit File* screen, select *Choose file* to specify the new contents of the file, then click *Edit File*.

# Edit File: CONTAINER\_1 : FILE\_1.txt



New File Contents \*

Choose File No file chosen

× CANCEL

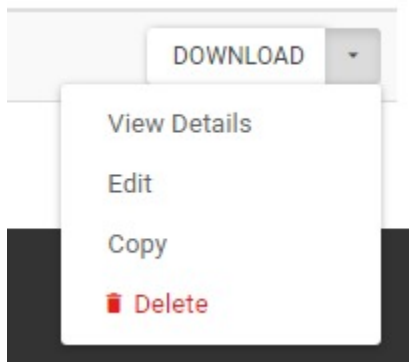
⬆ EDIT FILE

If successful, a success message will appear, and the file details will be updated.

**Warning:** Editing a file will remove any existing properties, apart from `Orig-Filename`, which will be set to the uploaded file name. See also: [Editing Object Metadata](#)

## 17.2.10 Copy a File

To copy a file, select the arrow next to *Download* to view the options, then click *Copy*.



On the *Copy Object* screen, enter the name of the destination container and destination file name, then click *Copy Object*.

# Copy Object: CONTAINER\_1/FILE\_1.txt



Destination Container \*

Destination Object \*

You can copy objects. You have to create destination container prior to copy.

You can specify folder by using '/' at destination object field. For example, if you want to copy object under the folder named 'folder1', you need to specify destination object like 'folder1/[your object name]'.

× CANCEL

↑ COPY OBJECT

If successful, a success message will appear, and the copied file will be listed.

## 17.2.11 Limitations

Some Swift features are unavailable through the GUI, and so must be performed using the *Python SDK* and/or the *CLI*. For example:

- *Viewing all metadata for your account, containers and objects*
- *Uploading and downloading multiple files simultaneously*
- *Setting metadata for containers and objects*
- *Downloading containers*
- *Downloading folders*
- *Uploading files larger than 5GiB*
- *Creating temporary URLs*

## 17.2.12 References

<https://docs.openstack.org/horizon/train/user/manage-containers.html>

## 17.3 Swift CLI

In order to use all the commands provided from Swift, it is recommended that you have the following python package installed:

```
pip install python-swiftclient
```

**Note:** See *Using OpenStack Command-line Interface* for how to set up the Openstack command line client. This is the preferred client, but in some cases the Swift client provides coverage that is not yet supported.

---

To test whether the Swift client have been installed successfully and we can access Swift commands, we can try to list containers in our current project:

```
swift list
```

This should return nothing if there are no containers in your project, or a list of containers similar to below:

```
CONTAINER_1
CONTAINER_2
CONTAINER_3
```

We can also try to list containers using the Openstack client, to test that this has been installed successfully and that we can access Swift commands in the preferred manner:

```
openstack container list
```

This should return an empty line if there are no containers in your project, or a table containing the names of containers similar to the table below:

```
+-----+
| Name   |
+-----+
| CONTAINER_1 |
| CONTAINER_2 |
| CONTAINER_3 |
+-----+
```

### 17.3.1 Commands

We are now able to run commands for working with containers and objects. Below is a list of some of the commands that can be used:

```
# Openstack client commands are of the form:
openstack container <commands>
openstack object <commands>

# Containers
container create <container> # Create a new container
container delete <container> # Delete a container
container list # List containers
container save <container> # Save a container's contents locally
container set --property <key>=<value> <container> # Set a container's properties
container show <container> # Display a container's details
container unset --property <key> <container> # Unset a container's properties

# Objects
object create <container> <object> # Upload an object to a container
object delete <container> <object> # Delete an object from a container
```

(continues on next page)

(continued from previous page)

```
object list <container> # List objects in a container
object save <container> <object> # Save an object locally
object set --property <key>=<value> <container> <object> # Set an object's properties
object show <container> <object> # Display an object's details
object unset --property <key> <container> <object> # Unset an object's properties
```

#### # Object store

```
object store account set --property <key>=<value> # Set an account's properties
object store account show # Display an account's details
object store account unset --property <key> # Unset an account's properties
```

#### # Swift client commands are of the form:

```
swift <commands>
```

#### # Containers

```
delete <container> # Delete a container, including all objects within
download <container> # Save a container's contents locally
list # List containers
post <container> # Update a container's details, or create a container if it does not_
↳ exist
stat <container> # Display a container's details
```

#### # Objects

```
copy <container> <object> # Updates an object's metadata, or copies an object to a new_
↳ destination
delete <container> <object> # Delete an object from a container
download <container> <object> # Save an object locally
list <container> # List objects in a container
post <container> <object> # Update an object's details
stat <container> <file> # Display's an object's details
upload <container> <object> # Upload an object to a container
```

#### # Other

```
auth # Display authentication variables
capabilities # Displays cluster capabilities for an object
delete --all # Deletes everything in the account
download --all # Downloads everything in the account
post # Update an account's details
stat # Display information for the account
tempurl <method> <time> <path> <key> # Generates a temporary URL for an object
```

**Warning:** Equivalent commands using the Openstack and Swift clients may differ in behaviour. For example, `openstack container delete <container>` will fail unless the container is empty or `-r` is used, whereas `swift delete <container>` will delete the contents of the container, as well as the container itself.

Further details and options can be seen using `openstack container <command> --help`, `openstack object <command> --help` and `swift <command> --help`.

### 17.3.2 Listing Files

To list objects in CONTAINER\_1:

```
openstack object list CONTAINER_1

# This should return a table similar to:
+-----+
| Name           |
+-----+
| FILE_1.txt     |
| FILE_2.txt     |
| FOLDER_1/FILE_1.txt |
+-----+
```

### 17.3.3 Getting Information About Your Account, Containers and Files

To print information about your account:

```
swift stat

# This should return text similar to:
Account: v1
Containers: 3
Objects: 9
Bytes: 21
Objects in policy "default-placement-bytes": 0
Bytes in policy "default-placement-bytes": 0
Containers in policy "default-placement": 3
Objects in policy "default-placement": 9
Bytes in policy "default-placement": 21
Meta Temp-Url-Key: MYKEY
Meta Key: value
X-Timestamp: 1654612569.67835
X-Account-Bytes-Used-Actual: 20480
X-Trans-Id: tx0000000000000008f227e6-00629f6259-
↪218cd2a5a-default
X-Openstack-Request-Id: tx0000000000000008f227e6-00629f6259-
↪218cd2a5a-default
Accept-Ranges: bytes
Content-Type: text/plain; charset=utf-8
```

To print basic information about CONTAINER\_1:

```
openstack container show CONTAINER_1

# This should return a table similar to:
+-----+-----+
| Field      | Value  |
+-----+-----+
| account    | v1     |
| bytes_used | 18     |
+-----+-----+
```

(continues on next page)

(continued from previous page)

```
| container      | CONTAINER_1 |
| object_count  | 3           |
+-----+-----+
```

To print more detailed information about CONTAINER\_1:

```
swift stat CONTAINER_1

# This should return text similar to:
    Account: v1
    Container: CONTAINER_1
    Objects: 3
    Bytes: 18
    Read ACL:
    Write ACL:
    Sync To:
    Sync Key:
    X-Timestamp: 1652205950.32071
X-Container-Bytes-Used-Actual: 12288
    X-Storage-Policy: default-placement
    X-Storage-Class: STANDARD
    Last-Modified: Fri, 13 May 2022 20:04:16 GMT
    X-Trans-Id: tx000000000000000058f1bbc-006283a793-21529d22a-default
X-Openstack-Request-Id: tx000000000000000058f1bbc-006283a793-21529d22a-default
    Accept-Ranges: bytes
    Content-Type: text/plain; charset=utf-8
```

To print basic information about FILE\_1.txt in CONTAINER\_1:

```
openstack object show CONTAINER_1 FILE_1.txt

# This should return a table similar to:
+-----+-----+
| Field      | Value                                     |
+-----+-----+
| account    | v1                                       |
| container  | CONTAINER_1                             |
| content-length | 4                                       |
| content-type | text/plain                             |
| etag       | 0cbc6611f5540bd0809a388dc95a615b     |
| last-modified | Tue, 17 May 2022 13:50:29 GMT         |
| object     | FILE_1.txt                             |
| properties | Orig-Filename='FILE_1.txt'             |
+-----+-----+
```

To print further information about FILE\_1.txt in CONTAINER\_1:

```
swift stat CONTAINER_1 FILE_1.txt

# This should return text similar to:
    Account: v1
    Container: CONTAINER_1
    Object: FILE_1.txt
```

(continues on next page)

(continued from previous page)

```

    Content Type: text/plain
    Content Length: 4
    Last Modified: Tue, 17 May 2022 13:50:29 GMT
    ETag: 0cbc6611f5540bd0809a388dc95a615b
    Meta Orig-Filename: FILE_1.txt
    Accept-Ranges: bytes
    X-Timestamp: 1652795429.54108
    X-Trans-Id: tx0000000000000004ef007b-006283a908-218cd2a5a-default
    X-Openstack-Request-Id: tx0000000000000004ef007b-006283a908-218cd2a5a-default

```

### 17.3.4 Creating Containers

Containers can be created using:

```

openstack container create [-h] [-f {csv,json,table,value,yaml}] [-c COLUMN]
                             [--quote {all,minimal,none,nonnumeric}] [--noindent] [--
↪max-width <integer>]
                             [--fit-width] [--print-empty] [--sort-column SORT_COLUMN]
                             [--sort-ascending | --sort-descending]
                             <container-name> [<container-name> ...]

Create new container

positional arguments:
  <container-name>
                        New container name(s)

optional arguments:
  -h, --help            show this help message and exit

output formatters:
  output formatter options

  -f {csv,json,table,value,yaml}, --format {csv,json,table,value,yaml}
                        the output format, defaults to table
  -c COLUMN, --column COLUMN
                        specify the column(s) to include, can be repeated to show
↪multiple columns
  --sort-column SORT_COLUMN
                        specify the column(s) to sort the data (columns specified first
↪have a priority, non-existing columns are
                        ignored), can be repeated
  --sort-ascending      sort the column(s) in ascending order
  --sort-descending     sort the column(s) in descending order

```

For example:

```

openstack container create CONTAINER_1

# This should return a table similar to:
+-----+-----+-----+-----+-----+-----+

```

(continues on next page)

(continued from previous page)

account	container	x-trans-id
v1	CONTAINER_1	tx0000000000000000384233c-006273aa93-21531bd94-default

**Note:** Containers created using the Openstack client will not be publicly accessible. This can be changed via the GUI, or by *updating the container's metadata*.

### 17.3.5 Uploading Files

Objects can be uploaded into containers using:

```
openstack object create [-h] [-f {csv,json,table,value,yaml}] [-c COLUMN]
                        [--quote {all,minimal,none,nonnumeric}] [--noindent] [--max-
↵width <integer>]
                        [--fit-width] [--print-empty] [--sort-column SORT_COLUMN] [-
↵--sort-ascending | --sort-descending]
                        [--name <name>]
                        <container> <filename> [<filename> ...]
```

Upload object to container

```
positional arguments:
```

```
<container>  Container for new object
<filename>   Local filename(s) to upload
```

optional arguments:

```
-h, --help          show this help message and exit
--name <name>       Upload a file and rename it. Can only be used when uploading a
→single object
```

```
output formatters:
```

output formatter options

```
-f {csv,json,table,value,yaml}, --format {csv,json,table,value,yaml}
    the output format, defaults to table
-c COLUMN, --column COLUMN
    specify the column(s) to include, can be repeated to show
multiple columns
--sort-column SORT_COLUMN
    specify the column(s) to sort the data (columns specified first
have a priority, non-existing columns are
ignored), can be repeated
--sort-ascending
    sort the column(s) in ascending order
--sort-descending
    sort the column(s) in descending order
```

CSV Formatter:

```
--quote {all,minimal,none,nonnumeric}
```

(continues on next page)

(continued from previous page)

```

                                when to include quotes, defaults to nonnumeric

json formatter:
    --noindent                    whether to disable indenting the JSON

table formatter:
    --max-width <integer>        Maximum display width, <1 to disable. You can also use the CLIFF_
    ↪MAX_TERM_WIDTH environment variable, but the
                                parameter takes precedence.
    --fit-width                    Fit the table to the display width. Implied if --max-width_
    ↪greater than 0. Set the environment variable
                                CLIFF_FIT_WIDTH=1 to always enable
    --print-empty                Print empty table if there is no data to show.

```

Multiple files may be uploaded simultaneously by listing then after the container name:

```
openstack object create CONTAINER_1 FILE_1.txt FILE_2.txt
```

*# This should return a table similar to:*

```

+-----+-----+-----+
| object      | container | etag                                     |
+-----+-----+-----+
| FILE_1.txt  | CONTAINER_1 | ff22941336956098ae9a564289d1bf1b |
| FILE_2.txt  | CONTAINER_1 | 9c8c1df0ae41d9a418d596e7ddfebf3b |
+-----+-----+-----+

```

**Note:** The name of the object uploaded will include its relative local path, unless otherwise specified using the `--name` option. For example, if `./FOLDER_1/FILE_1.txt` is uploaded, it will be named `FOLDER_1/FILE_1.txt` in the container by default.

### 17.3.6 Creating Folders

Folders can be created when uploading a file. For example, `FOLDER_1` and `FOLDER_2` can be created with the following:

```
openstack object create CONTAINER_1 FOLDER_1/FOLDER_2/FILE_1.txt
```

*# This should return a table similar to:*

```

+-----+-----+-----+
| object                                | container | etag                                     |
+-----+-----+-----+
| FOLDER_1/FOLDER_2/FILE_1.txt         | CONTAINER_1 | 2205e48de5f93c784733ffcca841d2b5 |
+-----+-----+-----+

```

To create an empty folder in a container, a local empty folder can be uploaded using the Swift client:

```
swift upload CONTAINER_1 FOLDER_1/
```

*# This should return the name of the created folder:*  
FOLDER\_1/

---

**Note:** The trailing / must be included to create a folder in this way.

---

### 17.3.7 Saving Containers

The full contents of a container can be saved using the Openstack client. For example:

```
openstack container save CONTAINER_1
```

All files will be downloaded to your current directory, with directories implied by object names being created as necessary to recreate the structure.

For example, saving the following container would save `FILE_1.txt` in `./FOLDER_1`, which will be created if it does not exist:

```
openstack object list CONTAINER_1
```

```
+-----+
| Name           |
+-----+
| FOLDER_1/FILE_1.txt |
+-----+
```

**Warning:** Local files will be overwritten if files with the same name are downloaded.

However, this command will fail if folders exist as unique objects in the container. For example, `FOLDER_1/` in the following:

```
openstack object list CONTAINER_1
```

```
+-----+
| Name           |
+-----+
| FOLDER_1/      |
| FOLDER_1/FILE_1.txt |
+-----+
```

---

**Note:** This occurs if folders have been created using the *Create Folder* screen, rather than the *Upload File* screen with the GUI, or if `swift upload <container> <empty folder>` has been used. The choice of folder creation mechanism should not affect the file structure when downloading containers/files or viewing files through the GUI.

---

In this case, the Swift client must be used to save containers:

```
swift download CONTAINER_1
```

By default, this will save all files to the current directory, and, as before, any directories that do not exist will be created.

### 17.3.8 Deleting Files

Multiple objects can be deleted using:

```
openstack object delete CONTAINER_1 FILE_1.txt FILE_2.txt
```

This will return nothing if successful.

### 17.3.9 Saving Files

Individual files can be saved using the Openstack client. For example:

```
openstack object save CONTAINER_1 FILE_1.txt
```

Multiple files can be saved using the Swift client. For example:

```
swift download CONTAINER_1 FILE_1.txt FILE_2.txt
```

### 17.3.10 Saving Folders

The full contents of a folder can be saved by using the Swift client to download each file. For example:

```
swift download CONTAINER_1 FOLDER_1/FILE_1.txt FOLDER_1/FILE_2.txt
```

### 17.3.11 Deleting Folders

If a folder is not a unique object, but exists through file names, it can be deleted by deleting all files within the folder. For example, if `FILE_1.txt` and `FILE_2.txt` are the only files in `FOLDER_1`, the following will delete the folder:

```
openstack object delete CONTAINER_1 FOLDER_1/FILE_1.txt FOLDER_1/FILE_2.txt
```

If a folder is stored as a unique object, this can be deleted in the same way as a file:

```
openstack object delete CONTAINER_1 FOLDER_1/
```

However, this will not delete any files within the folder. To delete the folder, both the folder object and the folder contents must be deleted:

```
openstack object delete CONTAINER_1 FOLDER_1/ FOLDER_1/FILE_1.txt FOLDER_1/FILE_2.txt
```

### 17.3.12 Deleting Containers

A container can be deleted using:

```
openstack container delete CONTAINER_1
```

This will return nothing if successful. An error will be thrown if the container is not empty, unless the `-r` or `--recursive` options are used to delete all objects within the container at the same time.

### 17.3.13 Large Files

Swift does not allow objects larger than 5GiB, so larger files must be segmented. This must be done using the Swift client:

```
swift upload <container> <object> --segment-size <size>
# <size> is the maximum segment size in Bytes. For example, to upload segments no larger
↳ than 1GiB:
swift upload CONTAINER_1 FILE_1.txt --segment-size 1G
```

**Warning:** Attempts to upload large files through the GUI or the Openstack client will fail. This may not occur until after an attempt has been made to upload the file, which may take a significant length of time.

This will upload the segments into a separate container, by default named <container>\_segments, and create a “manifest” file describing the entire object in <container>.

**Note:** A Dynamic Large Object is created by default, but if --use-slo is included with segment-size, a Static Large Object will be created instead. This still allows concurrent upload of segments and downloads via a single object, but it does not rely on eventually consistent container listings.

The entire object can be downloaded via the manifest file as if it were any other file, through the GUI or using the Openstack client:

```
openstack object save CONTAINER_1 FILE_1.txt
```

To delete the entire object, the Swift client must be used. For example:

```
swift delete CONTAINER_1 FILE_1.txt
```

If successful, this will output the manifest file name, as well as the file names of each segment, all of which will have been deleted. The segments container must be deleted separately.

**Warning:** Attempting to delete a segmented file using `openstack object delete` will delete the manifest file, but not the segments. In this case, the folder containing the segments must be deleted manually, as described in the first example of *Deleting Folders*.

### 17.3.14 Copying Files

Multiple files can be copied within a container, or between containers, using the Swift client. For example, copying `FILE_1.txt` and `FILE_2.txt` from `CONTAINER_1` to `CONTAINER_2`:

```
swift copy --destination /CONTAINER_2 CONTAINER_1 FILE_1.txt FILE_2.txt
```

**Warning:** This will overwrite any destination files sharing the same name.

If successful, this will create any containers and folders specified that do not exist, and output `<file> copied to <destination>` for each file.

**Note:** The output will also include *created container* <container>, even for containers that already exist.

---

### 17.3.15 Editing Container Metadata

Multiple custom properties of a container can be added or overwritten simultaneously through repeated use of the `--property` option:

```
openstack container set --property KEY_1=VALUE_1 --property KEY_2=VALUE_2 CONTAINER_1
```

**Note:** Underscores ('\_') in the property key will be converted to dashes ('-').

---

These properties will be listed when printing information about the container:

```
openstack container show CONTAINER_1

+-----+-----+
| Field          | Value                                |
+-----+-----+
| account        | v1                                  |
| bytes_used     | 8                                   |
| container      | CONTAINER_1                        |
| object_count   | 3                                   |
| properties     | Key-1='VALUE_1', Key-2='VALUE_2'   |
+-----+-----+
```

Custom properties may also be removed from a container:

```
openstack container unset --property KEY_1 --property KEY_2 CONTAINER_1
```

Setting system metadata, such making a container public, can be done through the Swift client using:

```
swift post <container> --read-acl ".r:*,.rlistings"
```

**Warning:** The above command will allow the contents of a container to be viewed by anyone, with no authentication required.

Access can instead be shared with specific groups. For example: <project-id>:<user-id>, <project-id>:\*, \*:<user-id> \*:\*, or <role\_name>. User IDs, rather than names, should be used, as names are not globally unique.

Similarly, containers can be made private using:

```
swift post <container> --read-acl ""
```

**Note:** Container ACLs are stored in the *X-Container-Write* and *X-Container-Read* metadata, but are set by `--write-acl` and `--read-acl` respectively.

Write access grants the ability to perform PUT, POST and DELETE operations on objects within a container, but not POST or DELETE operations on the container itself.

Read access grants the ability to perform GET and HEAD operations on objects within a container, but access to privileged metadata such as *X-Container-Sync-Key* is not granted.

### 17.3.16 Editing Object Metadata

As for containers, multiple custom properties for object may be set simultaneously:

```
openstack object set --property KEY_1=VALUE_1 --property KEY_2=VALUE_2 CONTAINER_1 FILE_1.txt
```

**Warning:** Any existing object properties that are not listed will be removed, including the default *Orig\_Filename* property.

**Note:** Underscores ('\_') in the property key will be converted to dashes ('-').

These properties will be listed when printing information about the object:

```
openstack object show CONTAINER_1 FILE_1.txt
```

Field	Value
account	v1
container	CONTAINER_1
content-length	4
content-type	text/plain
etag	0cbc6611f5540bd0809a388dc95a615b
last-modified	Tue, 17 May 2022 17:35:29 GMT
object	FILE_1.txt
properties	Key-1='VALUE_1', Key-2='VALUE_2'

Custom properties may also be removed from objects:

```
openstack object unset --property KEY_1 --property KEY_2 CONTAINER_1 FILE_1.txt
```

### 17.3.17 Creating a Temporary URL

Secret keys used in the cryptographic signature for temporary URLs can be created using the Swift client. For example, to set the secret key to *MYKEY*:

```
swift post -m "Temp-URL-Key:MYKEY"
```

**Note:** Two secret key values per account, and two per container can be stored.

A temporary URL for a Swift object can then be generated using the Swift client, typically to allow GET or PUT access. For example:

```
swift tempurl GET 1000 https://s3.echo.stfc.ac.uk/swift/v1/CONTAINER_1/FILE_1.txt MYKEY

# This should return a URL similar to:
https://s3.echo.stfc.ac.uk/swift/v1/CONTAINER_1/FILE_1.txt?temp_url_sig=?temp_url_
↪ sig=da39a3ee5e6b4b0d3255bfef95601890afd80709&temp_url_expires=1323479485
```

---

**Note:** The URL returned includes an ampersand, so must be enclosed in quotation marks in a command shell.

---

### 17.3.18 References

<https://docs.openstack.org/python-openstackclient/train/cli/command-objects/container.html?>

<https://docs.openstack.org/python-openstackclient/train/cli/command-objects/object.html>

<https://docs.openstack.org/python-openstackclient/train/cli/decoder.html#swift-cli>

<https://docs.openstack.org/python-swiftclient/train/cli/index.html>

[https://docs.openstack.org/swift/train/overview\\_large\\_objects.html](https://docs.openstack.org/swift/train/overview_large_objects.html)

[https://docs.openstack.org/swift/train/overview\\_acl.html](https://docs.openstack.org/swift/train/overview_acl.html)

[https://docs.openstack.org/swift/train/api/temporary\\_url\\_middleware.html#secret-keys](https://docs.openstack.org/swift/train/api/temporary_url_middleware.html#secret-keys)

## 17.4 Object Storage using the Python SDK

---

**Note:** See *Python SDK* for how to set up using the Python SDK.

---

### 17.4.1 Containers

Containers for an account can be interacted with using a `Connection` instance, `conn`, and the Object Store service. Some examples are given below.

---

**Note:** Containers can typically be passed into functions using either a `Container` instance or the container name as a string.

---

Listing containers for an account:

```
for cont in conn.object_store.containers():
    print(cont)
```

Creating a new container:

```
cont = conn.object_store.create_container("CONTAINER_1")
```

Deleting a container using its name:

```
conn.object_store.delete_container("CONTAINER_1")
```

---

**Note:** The container must be empty before deletion. This command will not raise an error if the container specified is not found unless `ignore_missing` is set to `False`.

---

Getting metadata for a container by passing a `Container` instance:

```
cont = conn.object_store.get_container_metadata(cont)
print(cont)
```

Setting system metadata for a container, such as the read and write access, as well as an example custom property, by passing a `Container` instance:

```
conn.object_store.set_container_metadata(cont, write_ACL="example_project:example_user",
↪read_ACL=".r:*,.rlistings", example_key="example_value")
```

---

**Note:** System metadata is set using system keys: `content_type`, `is_content_type_detected`, `versions_location`, `read_ACL`, `write_ACL`, `sync_to`, `sync_key`.

---

Deleting metadata for a container by passing a `Container` instance:

```
conn.object_store.delete_container_metadata(cont, ["example_key", "write_ACL", "read_ACL
↪"])
```

---

**Note:** Alternatively, `set_container_metadata` can be used with values set to `""` to delete custom and system metadata.

---



---

**Note:** When setting and deleting custom metadata, capitalisation is ignored, and when deleting custom metadata, `'-` and `'_'` are treated equivalently. This is not the case for system metadata.

---

## 17.4.2 Objects

Objects in a container can be interacted with using a `Connection` instance, `conn`, and the Object Store service. Some examples are given below.

---

**Note:** Similarly to interacting with containers, objects can typically be specified using either an `Object` instance or the object and container names as strings.

---

Listing objects in a container by passing the container name:

```
objs = conn.object_store.objects("CONTAINER_1")
for obj in objs:
    print(obj)
```

In the example above, `objs` is a generator object. Specific `Object` instances can be obtained from this in a number of ways, such as list comprehension:

```
obj_1 = [obj for obj in objs if obj.name=="FILE_1.txt"][0]
```

Objects can also be accessed directly using the container name and file name to return an Object instance:

```
obj_2 = conn.object_store.get_object("FILE_1.txt", "CONTAINER_1")
```

Equivalently:

```
obj_2 = conn.object_store.get_object_metadata("FILE_1.txt", "CONTAINER_1")
```

---

**Note:** The Object instance returned by the two examples above (obj\_2) differs slightly to that obtained using `conn.object_store.objects()` (obj\_1).

For example, the file name can be obtained via the `name` or `id` attributes of obj\_1, but only the `id` attribute of obj\_2. However, obj\_2 includes metadata not included in obj\_1, such as `accept-ranges` and `x-timestamp`.

---

Specific objects can also be accessed via a `Connection` instance by passing the container name and file name. This will return a tuple, containing (headers, body) for the object specified:

```
obj_tuple = conn.get_object('CONTAINER_1', 'FILE_1.txt')
```

Similarly, using a `Connection` instance, container name and file name, a `Response` object can be returned, which stores the object headers and content as attributes:

```
response = conn.get_object_raw('CONTAINER_1', 'FILE_1.txt')
```

Getting metadata for a container using an Object instance (in the form of either obj\_1 or obj\_2):

```
obj = conn.object_store.get_object_metadata(obj)
print(obj)
```

---

**Note:** If an object in the form of obj\_1 is passed to `get_object_metadata`, the object returned will include all the attributes of both obj\_1 and obj\_2.

---

Downloading an object's contents using an Object instance (in the form of either obj\_1 or obj\_2):

```
file_1 = conn.object_store.download_object(obj)
```

Alternatively, downloading contents using the Response object from above:

```
file_1 = response.content
```

In the two examples above, file\_1 will store the file contents as a bytes object. This can be written out in a number of ways, such as:

```
with open("SAVED_FILE_1.txt", "wb") as binary_file:
    binary_file.write(file_1)
```

Saving contents directly, without storing an intermediate Object or Response object:

```
conn.get_object('CONTAINER_1', 'FILE_1.txt', outfile="SAVED_FILE_1.txt")
```

Uploading a new object:

```
new_obj = conn.object_store.upload_object(container="CONTAINER_1",
                                          name="FILE_1.txt",
                                          data="Hello, world!")
```

Deleting an object using the container and file names:

```
conn.object_store.delete_object("FILE_1.txt", container="CONTAINER_1")
```

**Note:** An error will not be raised if the object specified is not found unless `ignore_missing` is set to `False`.

Setting system and custom metadata for an object by passing an `Object` instance:

```
conn.object_store.set_object_metadata(obj, delete_after="3000", example_key="example_
↪value")
```

**Note:** System metadata is set using system keys: `content_type`, `content_encoding`, `content_disposition`, `delete_after`, `delete_at`, `is_content_type_detected`.

`delete_at` is also set automatically by setting `delete_after`.

Deleting custom metadata for an object using the file and container names:

```
conn.object_store.delete_object_metadata("FILE_1.txt", "CONTAINER_4", ["example-key"])
```

**Note:** Alternatively, `set_object_metadata` can be used with values set to `""` to delete custom metadata. However, neither option will delete system metadata.

**Note:** When deleting custom metadata, the key should be in lower case, and underscores, `'_'`, in the original key name should be replaced with dashes, `'-'`.

### 17.4.3 swiftclient

An alternative to `openstacksdk` is `swiftclient`, which comprises a command line tool (see *Swift CLI*) and two separate APIs, *SwiftService* and *Connection*, for accessing swift programmatically.

This can be installed using:

```
pip install python-swiftclient
```

## SwiftService

Below are two examples to illustrate the use of the `swiftclient.SwiftService` API.

---

**Note:** There are several authentication mechanisms available, including setting environment variables that are automatically passed to your *SwiftService* instance. See *Using OpenStack Command-line Interface* for how to set these up.

---

Listing containers for your account:

```
import logging
from swiftclient.service import SwiftService, SwiftError

logging.basicConfig(level=logging.ERROR)
logging.getLogger("requests").setLevel(logging.CRITICAL)
logging.getLogger("swiftclient").setLevel(logging.CRITICAL)
logger = logging.getLogger(__name__)

with SwiftService() as swift:
    try:
        list_parts_gen = swift.list()
        for page in list_parts_gen:
            if page["success"]:
                for item in page["listing"]:
                    i_name = item["name"]
                    i_size = int(item["bytes"])
                    i_count = int(item["count"])
                    print(f"{i_name} [size: {i_size}] [count: {i_count}]")
    except SwiftError as e:
        logger.error(e.value)
```

Listing and downloading all text files in a container:

```
import logging
from swiftclient.service import SwiftService, SwiftError

logging.basicConfig(level=logging.ERROR)
logging.getLogger("requests").setLevel(logging.CRITICAL)
logging.getLogger("swiftclient").setLevel(logging.CRITICAL)
logger = logging.getLogger(__name__)

def is_txt(obj):
    return (
        obj["name"].lower().endswith('.txt') or
        obj["content_type"] == 'text/plain'
    )

container = "CONTAINER_1"

with SwiftService() as swift:
    try:
        list_options = {"prefix": "archive_2016-01-01/"}
```

(continues on next page)

(continued from previous page)

```

for page in list_parts_gen:
    if page["success"]:
        objects = [
            obj["name"] for obj in page["listing"] if is_txt(obj)
        ]
        for down_res in swift.download(
            container=container,
            objects=objects):
            if down_res['success']:
                print(f'{down_res['object']}' downloaded")
            else:
                print(f'{down_res['object']}' download failed")
        else:
            raise page["error"]
except SwiftError as e:
    logger.error(e.value)

```

**Warning:** The `content_type` object key may not exist for objects, such as placeholders, which can lead to errors if the above code is run.

## Connection

Below are two examples to illustrate the use of the `swiftclient.Connection` API.

**Note:** There are a number of kwarg combinations that can be used when creating a *Connection* instance for authentication.

Listing the response headers and containers for your account:

```

from swiftclient.client import Connection
from keystoneauth1 import session
from keystoneauth1.identity import v3

# Create a password auth plugin
auth = v3.Password(auth_url='https://openstack.stfc.ac.uk:5000/v3',
                   username='example_user',
                   password='example_password',
                   project_id='example_id',
                   user_domain_name="example_domain_name")

# Create session
keystone_session = session.Session(auth=auth)

# Create swiftclient Connection
conn = Connection(session=keystone_session)

resp_headers, containers = conn.get_account()
print(f"Response headers: {resp_headers}")
for container in containers:

```

(continues on next page)

(continued from previous page)

```
c_name = container["name"]
c_count = int(container["count"])
c_size = int(container["bytes"])
print(f"{c_name} [size: {c_size}] [count: {c_count}]")
```

Deleting an object:

```
from swiftclient.client import Connection, ClientException
from keystoneauth1 import session
from keystoneauth1.identity import v3

# Create a password auth plugin
auth = v3.Password(auth_url='https://openstack.stfc.ac.uk:5000/v3',
                   username='example_user',
                   password='example_password',
                   project_id='example_id',
                   user_domain_name="example_domain_name")

# Create session
keystone_session = session.Session(auth=auth)

# Create swiftclient Connection
conn = Connection(session=keystone_session)

obj = 'FILE_1.txt'
container = 'CONTAINER_1'
try:
    conn.delete_object(container, obj)
    print("Successfully deleted the object")
except ClientException as e:
    print(f"Failed to delete the object with error: {e}")
```

## 17.4.4 References

<https://docs.openstack.org/openstacksdk/train/>

[https://docs.openstack.org/openstacksdk/train/user/resources/object\\_store/v1/container.html](https://docs.openstack.org/openstacksdk/train/user/resources/object_store/v1/container.html)

[https://docs.openstack.org/openstacksdk/train/user/resources/object\\_store/v1/obj.html](https://docs.openstack.org/openstacksdk/train/user/resources/object_store/v1/obj.html)

[https://docs.openstack.org/openstacksdk/train/user/proxies/object\\_store.html](https://docs.openstack.org/openstacksdk/train/user/proxies/object_store.html)

<https://docs.openstack.org/openstacksdk/train/user/connection.html>

[https://docs.openstack.org/openstacksdk/train/user/guides/object\\_store.html](https://docs.openstack.org/openstacksdk/train/user/guides/object_store.html)

<https://docs.openstack.org/python-swiftclient/train/introduction.html>

## 18.1 OpenStack Magnum: Container-as-a-Service

**Warning:** Deprecated

Magnum is deprecated and will be replaced in the future with Cluster API.

Magnum is the Container-as-a-Service for OpenStack and can be used to create and launch clusters.

The clusters Magnum supports are:

- Kubernetes
- Swarm
- Mesos

Magnum uses the following OpenStack Components:

- **Keystone:** for multi tenancy
- **Nova compute:** computing service
- **Heat:** virtual application deployment service
- **Neutron:** networking
- **Glance:** virtual machine image
- **Cinder:** volume service

Magnum uses Cluster Templates to define the desired cluster, and passes the template for the cluster to Heat to create the user's cluster.

### 18.1.1 python-magnumclient

To use commands for creating or managing clusters using Magnum in the command line, you will need to install the Magnum CLI. This can be done using pip:

```
pip install python-magnumclient
```

To test that we can run OpenStack commands for container orchestration engines (coe), we can run the following command:

```
openstack coe cluster list #list clusters in the project
```

This should return an empty line if there are no clusters in the project, or a table similar to the following:

```

+-----+-----+-----+-----+-----+-----+
↪-----+-----+
| uuid      | name                | keypair      | node_count | master_count | ↪
↪status      | health_status |
+-----+-----+-----+-----+-----+-----+
↪-----+-----+
| UUID_1    | test-1              | mykeypair    | 1          | 1            | ↪
↪CREATE_COMPLETE | UNKNOWN          |
| UUID_2    | kubernetes-cluster-test-2 | mykeypair    | 1          | 1            | ↪
↪CREATE_COMPLETE | UNKNOWN          |
+-----+-----+-----+-----+-----+-----+
↪-----+-----+

```

Other commands available from python-magumclient include:

```

openstack coe <commands>

coe cluster config # download cluster config to current directory
coe cluster create # create a cluster
coe cluster delete # delete a cluster
coe cluster list # list clusters
coe cluster resize # resize cluster
coe cluster show # view details of cluster
coe cluster template create # create a cluster template
coe cluster template delete # delete a cluster template - can only be deleted if there
↪are no clusters using the template
coe cluster template list # list templates
coe cluster template show # view details of cluster template
coe cluster template update # update a cluster template
coe cluster update # update a cluster e.g node count
coe cluster upgrade # upgrade a cluster e.g upgrade Kubernetes version in cluster
coe nodegroup create # create a nodegroup
coe nodegroup delete # delete a nodegroup
coe nodegroup list # list nodegroups in cluster
coe nodegroup show # view details of a nodegroup in a cluster
coe nodegroup update # update a nodegroup

```

To view details of a cluster:

```
openstack coe cluster view <cluster-uuid>
```

*#This should return a table similar to the table below:*

```

+-----+-----+-----+-----+-----+-----+
↪-----+-----+
↪-----+
| Field                | Value                | ↪
↪
↪
↪

```

(continues on next page)

(continued from previous page)

```

+-----+
+-----+
| status          | CREATE_COMPLETE
|
| health_status   | UNKNOWN
|
| cluster_template_id | e186c6e2-dd47-4df0-ac3f-3eb46e64cb3d
|
| node_addresses  | ['10.0.0.163']
|
| uuid            | 27cdcad8-375f-4d4f-a186-8fa99b80c5c5
|
| stack_id        | e881d058-db91-4de6-9527-193eecebd05d
|
| status_reason   | None
|
| created_at      | 2020-09-07T15:39:32+00:00
|
| updated_at      | 2020-09-07T15:52:54+00:00
|
| coe_version     | v1.14.3
|
| labels          | {'auto_healing': 'true', 'kube_tag': 'v1.14.3', 'heat_container_
agent_tag': 'train-stable-3', 'kube_dashboard_enabled': '1', 'ingress_controller':
'traefik'}
| labels_overridden |
|
| labels_skipped   |
|
| labels_added     |
|
| fixed_network    | None
|
| fixed_subnet     | None
|
| floating_ip_enabled | False

```

(continues on next page)

(continued from previous page)

```

↪      |
↪ | faults          |
↪      |
↪      |
↪ | keypair         | mykeypair
↪      |
↪      |
↪ | api_address     | https://10.0.0.212:6443
↪      |
↪      |
↪ | master_addresses | ['10.0.0.212']
↪      |
↪      |
↪ | master_lb_enabled |
↪      |
↪      |
↪ | create_timeout   | 60
↪      |
↪      |
↪ | node_count       | 1
↪      |
↪      |
↪ | discovery_url     | https://discovery.etcd.io/31c1d9cf44cf4fda5710946d57980bb1
↪      |
↪      |
↪ | master_count      | 1
↪      |
↪      |
↪ | container_version | 1.12.6
↪      |
↪      |
↪ | name             | kubernetes-cluster-test-1
↪      |
↪      |
↪ | master_flavor_id  | c1.medium
↪      |
↪      |
↪ | flavor_id         | c1.medium
↪      |
↪      |
↪ | health_status_reason | {'api': 'The cluster kubernetes-cluster-test-1 is not_
↪ accessible.'}
↪      |
↪ | project_id        | PROJECT_ID
↪      |
↪ +-----+
↪ -----
↪ -----+

```

To view the list of cluster templates:

```
openstack coe cluster template list
```

*# This should return a list of cluster templates in the project*

```
+-----+-----+
| uuid                                | name                                |
+-----+-----+
| e186c6e2-dd47-4df0-ac3f-3eb46e64cb3d | kubernetes-v1_14_3                |
| 0bd1232d-06f2-42ca-b6d5-c27e57f26c3c | kubernetes-ha-master-v1_14_3      |
| a07903d0-aecf-4f15-a35f-f4fd74060e2f | coreos-kubernetes-v1_14_3         |
+-----+-----+
```

To view the details of a specific template:

```
openstack coe cluster template show <cluster-template-uuid>
```

*#This will return a table similar to:*

```
+-----+-----+
| Field                                | Value                                |
+-----+-----+
| insecure_registry                    | -                                    |
| labels                               | {'kube_tag': 'v1.14.3', 'kube_dashboard_enabled': '1', 'heat_
| container_agent_tag': 'train-stable-3', 'auto_healing': 'true', 'ingress_controller':
| 'traefik'} |
| updated_at                           | -                                    |
| floating_ip_enabled                  | False                               |
| fixed_subnet                         | -                                    |
| master_flavor_id                     | c1.medium                           |
| uuid                                 | e186c6e2-dd47-4df0-ac3f-3eb46e64cb3d |
| no_proxy                             | -                                    |
| https_proxy                          | -                                    |
```

(continues on next page)

(continued from previous page)

```

↪      |
| tls_disabled          | False
↪
↪      |
| keypair_id           | -
↪
↪      |
| public               | True
↪
↪      |
| http_proxy           | -
↪
↪      |
| docker_volume_size   | 3
↪
↪      |
| server_type          | vm
↪
↪      |
| external_network_id  | External
↪
↪      |
| cluster_distro       | fedora-atomic
↪
↪      |
| image_id             | cf37f7d0-1d6b-4aab-a23b-df58542c59cb
↪
↪      |
| volume_driver        | -
↪
↪      |
| registry_enabled     | False
↪
↪      |
| docker_storage_driver | devicemapper
↪
↪      |
| apiserver_port       | -
↪
↪      |
| name                 | kubernetes-v1_14_3
↪
↪      |
| created_at           | 2020-09-07T07:17:13+00:00
↪
↪      |
| network_driver       | flannel
↪
↪      |
| fixed_network        | -
↪
↪      |

```

(continues on next page)

(continued from previous page)

coe	kubernetes	
↪		
↪		
flavor_id	c1.medium	
↪		
↪		
master_lb_enabled	False	
↪		
↪		
dns_nameserver	8.8.8.8	
↪		
↪		
hidden	False	
↪		
↪		
+-----+		
↪-----		
↪-----+		

To delete a cluster or cluster template:

**Note:** Cluster Templates can only be deleted if there are no clusters using the template.

```
# To delete a template
openstack coe cluster template delete <cluster-template-id>

# To delete a cluster
openstack coe cluster delete <cluster-id>
```

## 18.1.2 Creating Clusters

Clusters can be created using:

- OpenStack CLI
- Horizon Web UI
- Heat Templates: using the resources `OS::Magnum::ClusterTemplate` and `OS::Magnum::Cluster`

The documentation **Create A Kubernetes Cluster** has examples for handling cluster templates and creating a Kubernetes cluster in the command line.

### Create a Cluster using OpenStack CLI

#### Create A Cluster Template

To create a cluster template, we can use the following command:

```
openstack coe cluster template create [-h] [-f {json,shell,table,value,yaml}] [-c   
↪ COLUMN] [--noindent] [--prefix PREFIX]   
                                     [--max-width <integer>] [--fit-width] [--print-   
↪ empty] --coe <coe> --image <image>
```

(continues on next page)

(continued from previous page)

```

↪keypair <keypair>]
↪subnet <fixed-subnet>]
↪driver <volume-driver>]
↪<flavor>]
↪volume-size <docker-volume-size>]
↪driver>] [--http-proxy <http-proxy>]
↪proxy>]
↪...>] [--tls-disabled] [--public]
↪type>] [--master-lb-enabled]
↪disabled] [--hidden] [--visible]
--external-network <external-network> [--
[--fixed-network <fixed-network>] [--fixed-
[--network-driver <network-driver>] [--volume-
[--dns-nameserver <dns-nameserver>] [--flavor
[--master-flavor <master-flavor>] [--docker-
[--docker-storage-driver <docker-storage-
[--https-proxy <https-proxy>] [--no-proxy <no-
[--labels <KEY1=VALUE1,KEY2=VALUE2;KEY3=VALUE3.
[--registry-enabled] [--server-type <server-
[--floating-ip-enabled] [--floating-ip-
<name>

```

<name>: Name of the ClusterTemplate to create. The name does not have to be unique but the template UUID should be used to select a ClusterTemplate if more than one template has the same name.

<coe>: Container Orchestration Engine to use. Supported drivers are: **kubernetes, swarm, mesos**.

<image>: Name or UUID of the base image to boot servers for the clusters.

#### Images which OpenStack Magnum supports:

COE	os_distro
Kubernetes	fedora-atomic, coreos
Swarm	fedora-atomic
Mesos	ubuntu

<keypair>: SSH keypair to configure in servers for ssh access. The login name is specific to the cluster driver.

- fedora-atomic: `ssh -i <private-key> fedora@<ip-address>`
- coreos: `ssh -i <private-key> core@<ip-address>`

`external-network <external-network>`: name or ID of a Neutron network to provide connectivity to the external internet.

`--public` Access to a ClusterTemplate is, by default, limited to admin, owner or users within the same tenant as the owners. Using this flag makes the template accessible by other users. Default is not public

`server-type <server-type>`: Servers can be VM or bare metal (bm). The default is vm.

`network-driver <network-driver>` Name of a network driver for providing networks for the containers - this is different and separate from the Neutron network for the cluster. Drivers that Magnum supports:

COE	Network Driver	Default
Kubernetes	flannel, calico	flannel
Swarm	docker, flannel	flannel
Mesos	docker	docker

**Note:** For Kubernetes clusters, we are using the flannel network driver.

**dns-nameserver** <dns-nameserver>: The DNS nameserver for the servers and containers in the cluster to use. The default is 8.8.8.8.

**flavor** <flavor>: flavor to use for worker nodes. The default is m1.small. Can be overridden at cluster creation.

**master-flavor** <master-flavor>: flavor for master nodes. Default is m1.small. Can be overridden at cluster creation.

**http-proxy** <http-proxy>: The IP address for a proxy to use when direct http access from the servers to sites on the external internet is blocked. The format is a URL including a port number. The default is None.

**https-proxy** <https-proxy>: The IP address for a proxy to use when direct https access from the servers to sites on the external internet is blocked. The format is a URL including a port number. The default is None.

**no-proxy** <no-proxy>: When a proxy server is used, some sites should not go through the proxy and should be accessed normally. In this case, you can specify these sites as a comma separated list of IPs. The default is None.

**docker-volume-size** <docker-volume-size>: If specified, container images will be stored in a cinder volume of the specified size in GB. Each cluster node will have a volume attached of the above size. If not specified, images will be stored in the compute instances local disk. For the devicemapper storage driver, must specify volume and the minimum value is 3GB. For the overlay and overlay2 storage driver, the minimum value is 1GB or None(no volume). This value can be overridden at cluster creation.

**docker-storage-driver** <docker-storage-driver>: The name of a driver to manage the storage for the images and the containers writable layer. The default is devicemapper.

**labels** <KEY1=VALUE1,KEY2=VALUE2;KEY3=VALUE3>: Arbitrary labels in the form of key=value pairs. The accepted keys and valid values are defined in the cluster drivers. They are used as a way to pass additional parameters that are specific to a cluster driver. The value can be overridden at cluster creation.

**--tls-disabled** Transport Layer Security (TLS) is normally enabled to secure the cluster. The default is TLS enabled.

**--registry-enabled** Docker images by default are pulled from the public Docker registry, but in some cases, users may want to use a private registry. This option provides an alternative registry based on the Registry V2: Magnum will create a local registry in the cluster backed by swift to host the images. Refer to Docker Registry 2.0 for more details. The default is to use the public registry.

**--master-lb-enabled** Since multiple masters may exist in a cluster, a load balancer is created to provide the API endpoint for the cluster and to direct requests to the masters. As we have Octavia enabled, Octavia would create these load balancers. The default is master load balancers are created.

## Create a Cluster

We can create clusters using a cluster template from our template list. To create a cluster, we use the command:

```
openstack coe cluster create --cluster-template <cluster-template>
                             --discovery-url <discovery-url>
                             --master-count <master-count>
                             --node-count <node-count>
                             --timeout <timeout>
                             --merge-labels
                             --master-lb-enabled
                             #The following options can be used to overwrite the same_
↪options in the cluster template
                             --docker-volume-size <docker-volume-size>
```

(continues on next page)

(continued from previous page)

```

--labels <KEY1=VALUE1,KEY2=VALUE2;KEY3=VALUE3...>
--keypair <keypair>
--master-flavor <master-flavor>
--flavor <flavor>
--fixed-network <fixed-network>
--fixed-subnet <fixed-subnet>
--floating-ip-enabled
--floating-ip-disabled
# To add labels to use with the template labels, we can
↪use:
--merge-labels
<name>

```

**Note:** It is recommended that to have master load balancers enabled, to use the `kubernetes-ha-master-v1_14_3` template, or create a new cluster template and include the flag `--master-lb-enabled`.

## Labels

Labels are used by OpenStack Magnum to define a range of parameters such as the Kubernetes version, enable autoscaling, enable autohealing, version of drain to use etc. Any labels included at cluster creation overwrite the labels in the cluster template. A table containing all of the labels which Magnum uses can be found here:

<https://docs.openstack.org/magnum/train/user/>

**Note:** For OpenStack Train release, Magnum only offers labels for installing Helm 2 and Tiller. However, Helm 3 can be installed onto the master node after the cluster has been created.

## Horizon Web Interface

Clusters can also be created using the Horizon Web Interface. Clusters and their templates can be found under the `Container Infra` section.

There are a few differences between the parameters which can be defined when creating a cluster using the CLI or Horizon Web UI. If you are using the Horizon web UI to create clusters, the **fixed network, fixed subnet, and floating ip enabled** can only be defined in the cluster template.

## Heat Templates

Clusters can also be created using a Heat template using the resources `OS::Magnum::ClusterTemplate` and `OS::Magnum::Cluster`.

This will instruct Heat to pass the resources to Magnum, which will pass a stack template to Heat to create a cluster - so two stacks are built in total.

## OS::Magnum::ClusterTemplate

```

resources:
  cluster_template:
    type: OS::Magnum::ClusterTemplate
    properties:
      #required
      coe: String # Container Orchestration Engine: kubernetes, swarm, mesos
      external_network: String # External neutron network or UUID to attach the cluster
      image: String # The image name/UUID to use as a base image for the cluster
      # optional
      dns_nameserver: String # DNS nameserver address, must be of type ip_addr
      docker_storage_driver: String # Docker storage driver: devicemapper, overlay
      docker_volume_size: Integer # Size in GB of docker volume, must be at least 1
      fixed_network: String # The fixed neutron network name or UUID to attach the
↪Cluster
      fixed_subnet: String # The fixed neutron subnet name or UUID to attach the
↪Cluster
      flavor: String # Flavor name or UUID to use when launching a cluster
      floating_ip_enabled: Boolean # True by default, determines whether a cluster
↪should have floating IPs
      http_proxy: String # http_proxy address to use for nodes in cluster
      https_proxy: String # https_proxy address to use for nodes in cluster
      keypair: String # SSH keypair to load into cluster nodes
      labels: {...} # labels in form of key=value pairs to associate with cluster
      master_flavor: String # flavor name or UUID to associate with the master node
      master_lb_enabled: Boolean # Defaults to true. Determines whether there should
↪be a load balancer for master nodes
      name: String # Template name
      network_driver: String # Name of driver to use for instantiating container
↪networks. Magnum uses pre-configured driver for specific COE by default
      no_proxy: String # A comma separated list of addresses for which proxies should
↪not be used in the cluster
      public: Boolean # Defaults to false. True makes the cluster template public.
↪Must have the permissions to publish templates in Magnum
      registry_enabled: Boolean # Defaults to false. Enable registry in the cluster
      server_type: String # Define server type to use. Defaults to vm. Allowed: vm, bm
      tls_disabled: Boolean # Disable TLS in the Cluster. Defaults to false
      volume_driver: String # Volume driver name for instantiating container volume.
↪Allowed: cinder, rexd
outputs:
  detailed-information:
    description: Detailed Information about the resource
    value: {get_attr: [cluster_template, show]}

```

## OS::Magnum::Cluster

```

resources:
  cluster:
    type: OS::Magnum::Cluster
    properties:
      # required
      cluster_template: String # Name or ID of cluster template
      # optional
      create_timeout: Integer # Timeout for creating cluster in minutes. Defaults to 60
      ↪minutes
      discovery_url: String # Specifies a custom discovery url for node discovery
      keypair: String # name of keypair. Uses keypair in template if keypair is not
      ↪defined here
      master_count: Integer # Number of master nodes, defaults to 1. Must be at least 1
      name: String # name of cluster
      node_count: Integer # Number of worker nodes, defaults to 1. Must be at least 1

outputs:

  api_address:
    description: Endpoint URL of COE API exposed to end-users
    value: {get_attr: [cluster, api_address]}
  cluster_template_id:
    description: UUID of cluster template
    value: {get_attr: [cluster, cluster_template_id]}
  coe_version:
    description: Version information of container engine in chosen COE in cluster
    value: {get_attr: [cluster, coe_version]}
  create_timeout:
    description: Timeout in minutes for cluster creation
    value: {get_attr: [cluster, create_timeout]}
  discovery_url:
    description: Custom discovery url for node discovery
    value: {get_attr: [cluster, discovery_url]}
  keypair:
    description: Name of keypair
    value: {get_attr: [cluster, keypair]}
  master_addresses:
    description: List of IPs for all master nodes
    value: {get_attr: [cluster, master_addresses]}
  master_count:
    description: Number of servers that serve as master for the cluster
    value: {get_attr: [cluster, master_count]}
  name:
    description: Name of the resource
    value: {get_attr: [cluster, name]}
  node_addresses:
    description: IP addresses of all worker nodes in the cluster
    value: {get_attr: [cluster, node_addresses]}
  node_count:
    description: Number of servers that will serve as node for the cluster

```

(continues on next page)

(continued from previous page)

```

    value: {get_attr: [cluster, node_count]}
show:
    description: Show detailed information about the cluster
    value: {get_attr: [cluster, show]}
stack_id:
    description: UUID of orchestration stack for this COE cluster
    value: {get_attr: [cluster, stack_id]}
status:
    description: Status of this COE cluster
    value: {get_attr: cluster, status}
    status_reason:
    description: The reason for the cluster current status
    value: {get_attr: cluster, status_reason}

```

## Example Template

For example, we could have the template `example.yaml` which outlines the template for a Kubernetes cluster and instructs heat to create a cluster using this template:

```

heat_template_version: 2018-08-31 #Train release

description: This is an example template to create a Kubernetes cluster.

parameters:
  keypair:
    type: string
    default: mykeypair

  image:
    type: string
    default: <IMAGE_ID>
    description: fedora-atomic

resources:
  stack_cluster_template:
    type: OS::Magnum::ClusterTemplate
    properties:
      coe: kubernetes
      dns_nameserver: 8.8.8.8
      docker_storage_driver: devicemapper
      docker_volume_size: 10
      external_network: External
      flavor: c1.medium
      floating_ip_enabled: false
      image: {get_param: image}
      labels: {'kube_tag': 'v1.14.3', 'kube_dashboard_enabled': '1', 'heat_container_
↪agent_tag': 'train-stable-3', 'auto_healing': 'true', 'ingress_controller': 'traefik'}
      master_flavor: c1.medium
      name: my-cluster-template
      network_driver: flannel
      registry_enabled: false

```

(continues on next page)

(continued from previous page)

```

server_type: vm
volume_driver: cinder
master_lb_enabled: false

test_cluster:
  type: OS::Magnum::Cluster
  properties:
    cluster_template: {get_resource: stack_cluster_template}
    create_timeout: 60
    keypair: {get_param: keypair}
    name: test-cluster
    node_count: 1
    master_count: 1

```

Then we can launch this stack using:

```
openstack stack create -t example.yaml test-cluster-stack
```

To delete a cluster created using example.yaml, delete the stack which was built by *example.yaml*:

```
openstack stack delete test-cluster-stack
```

## Accessing the Cluster

To access the cluster, add a floating IP to the master node and ssh using:

```

#For Fedora Atomic
ssh -i <private-key> fedora@<master-ip>

#For coreOS
ssh -i <private-key> core@<master-ip>

```

### 18.1.3 Upgrading Clusters

Rolling upgrades can be applied to Kubernetes Clusters using the command `openstack coe cluster upgrade <cluster-id> <new-template-id>`. This command can be used for upgrading the Kubernetes version or for upgrading the node operating system version.

**Note:** Downgrading is not supported

#### openstack coe cluster upgrade

```

openstack coe cluster upgrade --help
usage: openstack coe cluster upgrade [-h] [--max-batch-size <max_batch_size>] [--
nodegroup <nodegroup>] <cluster> cluster_template

```

Upgrade a Cluster

positional arguments:

(continues on next page)

(continued from previous page)

<code>&lt;cluster&gt;</code>	The name or UUID of cluster to update
<code>cluster_template</code>	The new cluster template ID will be upgraded to.

optional arguments:

<code>-h, --help</code>	show this <b>help</b> message and <b>exit</b>
<code>--max-batch-size &lt;max_batch_size&gt;</code>	The max batch size <b>for</b> upgrading each time.
<code>--nodegroup &lt;nodegroup&gt;</code>	The name or UUID of the nodegroup of current cluster.

## Example

This example will go through how to upgrade an existing cluster to use Kubernetes v1.15.7.

The cluster we will update has the following features:

Field	Value
status	UPDATE_COMPLETE
health_status	UNKNOWN
cluster_template_id	e186c6e2-dd47-4df0-ac3f-3eb46e64cb3d
node_addresses	['10.0.0.131', '10.0.0.8']
uuid	686f9fa1-eb56-4c23-9afd-67a79c283736
stack_id	80b6af23-8a14-4a44-bc62-b77d9eb6736b
status_reason	None
created_at	2020-11-16T12:46:28+00:00
updated_at	2020-11-27T11:47:32+00:00
coe_version	v1.15.7
labels	{'auto_healing_controller': 'draino', 'max_node_count': '4', 'kube_tag': 'v1.14.3', 'min_node_count': '1', 'ingress_controller': 'traefik', 'auto_healing_enabled': 'true', 'heat_container_agent_tag': 'train-stable-3', 'auto_scaling_enabled': 'true'}
labels_overridden	
labels_skipped	

(continues on next page)

(continued from previous page)

↳			
↳	labels_added		
↳	fixed_network	None	
↳	fixed_subnet	None	
↳	floating_ip_enabled	False	
↳	faults		
↳	keypair	mykeypair	
↳	api_address	https://10.0.0.117:6443	
↳	master_addresses	['10.0.0.201']	
↳	master_lb_enabled		
↳	create_timeout	60	
↳	node_count	2	
↳	discovery_url	https://discovery.etcd.io/6b47ff194fc4dcefb3a7d430d69e761c	
↳	master_count	1	
↳	container_version	1.12.6	
↳	name	k8s-cluster	
↳	master_flavor_id	c1.medium	
↳	flavor_id	c1.medium	
↳	health_status_reason	{'api': 'The cluster k8s-cluster is not accessible.'}	
↳	project_id	PROJECT_ID	
↳	+	-----+	
↳	+	-----+	

To upgrade the Kubernetes version for our cluster, we create a new template where we change the value of the label `kube_tag` from `v1.14.3` to `v1.15.7`

```
openstack coe cluster template create --coe kubernetes \
--image cf37f7d0-1d6b-4aab-a23b-df58542c59cb \
--external-network External \
--network-driver flannel \
--volume-driver cinder \
--dns-nameserver 8.8.8.8 \
--flavor c1.medium \
```

(continues on next page)

(continued from previous page)

```

--master-flavor c1.medium \
--docker-volume-size 10 \
--docker-storage-driver devicemapper \
--labels auto_healing_controller=draino,auto_
↪healing_enabled=true,heat_container_agent_tag=train-stable-3,ingress_
↪controller=traefik \
--labels auto_scaling_enabled=true,min_node_
↪count=1,max_node_count=4,kube_tag=1.15.7
--server_type vm
update-template

```

Then we apply the cluster upgrade to this cluster:

```

openstack coe cluster upgrade 686f9fa1-eb56-4c23-9afd-67a79c283736 <update-template-id>
# If the command is successful, the following message should be returned:

```

```

Request to upgrade cluster 686f9fa1-eb56-4c23-9afd-67a79c283736 has been accepted.

```

The cluster will then move into UPDATE\_IN\_PROGRESS state while the cluster updates the Kubernetes version. The cluster will move to UPDATE\_COMPLETE status when the upgrade is complete. We can verify that our cluster is using a different version of Kubernetes by using SSH to connect to the master node and running the following command:

```

$ kubectl version

Client Version: version.Info{Major:"1", Minor:"15", GitVersion:"v1.15.7", GitCommit:
↪"6c143d35bb11d74970e7bc0b6c45b6bfdffc0bd4", GitTreeState:"clean", BuildDate:"2019-12-
↪11T12:42:56Z", GoVersion:"go1.12.12", Compiler:"gc", Platform:"linux/amd64"}
Server Version: version.Info{Major:"1", Minor:"15", GitVersion:"v1.15.7", GitCommit:
↪"6c143d35bb11d74970e7bc0b6c45b6bfdffc0bd4", GitTreeState:"clean", BuildDate:"2019-12-
↪11T12:34:17Z", GoVersion:"go1.12.12", Compiler:"gc", Platform:"linux/amd64"}

$ sudo docker version

Client:
Version:      1.13.1
API version:  1.26
Package version: docker-1.13.1-68.git47e2230.fc29.x86_64
Go version:   go1.11.12
Git commit:   47e2230/1.13.1
Built:        Sat Aug 17 20:18:33 2019
OS/Arch:      linux/amd64

Server:
Version:      1.13.1
API version:  1.26 (minimum version 1.12)
Package version: docker-1.13.1-68.git47e2230.fc29.x86_64
Go version:   go1.11.12
Git commit:   47e2230/1.13.1
Built:        Sat Aug 17 20:18:33 2019
OS/Arch:      linux/amd64
Experimental: false

```

We can see that the Kubernetes and Docker version have been upgraded for our cluster.

### 18.1.4 Updating Clusters

Clusters can be modified using the command:

```
openstack coe cluster update [-h] [--rollback] <cluster> <op> <path=value> [<path=value>...]
```

Update a Cluster

positional arguments:

<cluster> The name or UUID of cluster to update  
 <op> Operations: one of 'add', 'replace' or 'remove'  
 <path=value> Attributes to add/replace or remove (only PATH is necessary on remove)

optional arguments:

-h, --help show this help message and exit  
 --rollback Rollback cluster on update failure.

The following table summarizes the possible changes that can be applied to the cluster.

Attribute	add	replace	remove
node_count	no	add/remove nodes	reset to default of 1
master_count	no	no	no
name	no	no	no
discovery_url	no	no	no

### 18.1.5 Resize a Cluster

The size of a cluster can be changed by using the following command:

```
openstack coe cluster resize [-h] [--nodes-to-remove <Server UUID>] [--nodegroup <nodegroup>] <cluster> node_count
```

Resize a Cluster

positional arguments:

<cluster> The name or UUID of cluster to update  
 node\_count Desired node count of the cluster.

optional arguments:

-h, --help show this help message and exit  
 --nodes-to-remove <Server UUID>  
 Server ID of the nodes to be removed. Repeat to addmore server ID  
 --nodegroup <nodegroup>  
 The name or UUID of the nodegroup of current cluster.

### 18.1.6 References:

<https://docs.openstack.org/magnum/train/user/>

[https://docs.openstack.org/heat/train/template\\_guide/openstack.html](https://docs.openstack.org/heat/train/template_guide/openstack.html)

<https://www.openstack.org/videos/summits/austin-2016/intro-to-openstack-magnum-with-kubernetes>

<https://object-storage-ca-ymq-1.vexxhost.net/swift/v1/6e4619c416ff4bd19e1c087f27a43eea/www-assets-prod/presentation-media/openstack-magnum-hands-on.pdf>

<https://www.openstack.org/videos/summits/denver-2019/container-use-cases-and-developments-at-the-cern-cloud>

## 18.2 Create a Kubernetes Cluster

**Warning:** Deprecated

Magnum is deprecated and will be replaced in the future with Cluster API.

Clusters are groups of resources (nova instances, neutron networks, security groups etc.) combined to function as one system. To do this, Magnum uses Heat to orchestrate and create a stack which contains the cluster.

This documentation will focus on how to create Kubernetes clusters using OpenStack Magnum.

### 18.2.1 Magnum CLI

Any commands for creating clusters using OpenStack Magnum begin with:

```
openstack coe <commands> <options>
```

In order to have the openstack commands for Magnum available to use through the CLI, you will need to install the python client for Magnum. This can be done using pip:

```
pip install python-magnumclient
```

Now the commands relating to the container orchestration engine, clusters, cluster templates are available on the command line.

### 18.2.2 Cluster Templates

Clusters can be created from templates which are passed through Heat. To view the list of cluster templates which are in your project, you can use the following command:

```
openstack coe cluster template list

# This should return either an empty line if there are no templates, or
# a table similar to the one below:
+-----+-----+
| uuid                                | name                                |
+-----+-----+
| e186c6e2-dd47-4df0-ac3f-3eb46e64cb3d | kubernetes-v1_14_3                |
| 0bd1232d-06f2-42ca-b6d5-c27e57f26c3c | kubernetes-ha-master-v1_14_3     |
```

(continues on next page)

(continued from previous page)

```
| a07903d0-aecf-4f15-a35f-f4fd74060e2f | coreos-kubernetes-v1_14_3 |
+-----+-----+
```

Templates can be created using the following command:

```
openstack coe cluster template create [-h] [-f {json,shell,table,value,yaml}] [-c
↪ COLUMN] [--noindent] [--prefix PREFIX]
                                [--max-width <integer>] [--fit-width] [--print-
↪ empty] --coe <coe> --image <image>
                                --external-network <external-network> [--
↪ keypair <keypair>]
                                [--fixed-network <fixed-network>] [--fixed-
↪ subnet <fixed-subnet>]
                                [--network-driver <network-driver>] [--volume-
↪ driver <volume-driver>]
                                [--dns-nameserver <dns-nameserver>] [--flavor
↪ <flavor>]
                                [--master-flavor <master-flavor>] [--docker-
↪ volume-size <docker-volume-size>]
                                [--docker-storage-driver <docker-storage-
↪ driver>] [--http-proxy <http-proxy>]
                                [--https-proxy <https-proxy>] [--no-proxy <no-
↪ proxy>]
                                [--labels <KEY1=VALUE1,KEY2=VALUE2;KEY3=VALUE3.
↪ ..>] [--tls-disabled] [--public]
                                [--registry-enabled] [--server-type <server-
↪ type>] [--master-lb-enabled]
                                [--floating-ip-enabled] [--floating-ip-
↪ disabled] [--hidden] [--visible]
                                <name>
```

Kubernetes Cluster Template:

```
$ openstack coe cluster template show e186c6e2-dd47-4df0-ac3f-3eb46e64cb3d
+-----+-----+
↪ -----+
↪ -----+
| Field          | Value
↪
↪ |
+-----+-----+
↪ -----+
| insecure_registry | -
↪
↪ |
| labels           | {'kube_tag': 'v1.14.3', 'kube_dashboard_enabled': '1', 'heat_
↪ container_agent_tag': 'train-stable-3', 'auto_healing': 'true', 'ingress_controller':
↪ 'traefik'} |
| updated_at       | -
↪
↪ |
```

(continues on next page)

(continued from previous page)

floating_ip_enabled	False
↳	
↳	
fixed_subnet	-
↳	
↳	
master_flavor_id	c1.medium
↳	
↳	
uuid	e186c6e2-dd47-4df0-ac3f-3eb46e64cb3d
↳	
↳	
no_proxy	-
↳	
↳	
https_proxy	-
↳	
↳	
tls_disabled	False
↳	
↳	
keypair_id	-
↳	
↳	
public	True
↳	
↳	
http_proxy	-
↳	
↳	
docker_volume_size	3
↳	
↳	
server_type	vm
↳	
↳	
external_network_id	External
↳	
↳	
cluster_distro	fedora-atomic
↳	
↳	
image_id	cf37f7d0-1d6b-4aab-a23b-df58542c59cb
↳	
↳	
volume_driver	-
↳	
↳	
registry_enabled	False
↳	
↳	
docker_storage_driver	devicemapper

(continues on next page)

(continued from previous page)

```

↪      |
↪ | apiserver_port      | -
↪      |
↪ | name                | kubernetes-v1_14_3
↪      |
↪ | created_at          | 2020-09-07T07:17:13+00:00
↪      |
↪ | network_driver      | flannel
↪      |
↪ | fixed_network       | -
↪      |
↪ | coe                 | kubernetes
↪      |
↪ | flavor_id           | c1.medium
↪      |
↪ | master_lb_enabled   | False
↪      |
↪ | dns_nameserver      | 8.8.8.8
↪      |
↪ | hidden              | False
↪      |
↪ +-----+
↪ -----
↪ -----+

```

### 18.2.3 Create a Kubernetes Cluster

We can create a Kubernetes cluster using one of the cluster templates that are available. To create a cluster, we use the command:

```

openstack coe cluster create --cluster-template <cluster-template> --discovery-url
↪ <discovery-url> --master-count <master-count> --node-count <node-count>
                                --timeout <timeout> --merge-labels
                                #The following options can be used to overwrite the same
↪ options in the cluster template
                                --docker-volume-size <docker-volume-size>
                                --labels <KEY1=VALUE1,KEY2=VALUE2;KEY3=VALUE3...> # --
↪ merge-labels flag will add labels to the ones provided by the template
                                --keypair <keypair>
                                --master-flavor <master-flavor>

```

(continues on next page)

(continued from previous page)

```

--flavor <flavor>
--fixed-network <fixed-network>
--fixed-subnet <fixed-subnet>
--floating-ip-enabled
--floating-ip-disabled
--master-lb-enabled
<name>

```

For example, consider a user that wants to create a cluster using the Kubernetes cluster template. They want the cluster to have:

- one master node
- one worker node
- their keypair mykeypair

```

openstack coe cluster create --cluster-template e186c6e2-dd47-4df0-ac3f-3eb46e64cb3d \
--keypair mykeypair \
--master-count 1 \
--node-count 1 \
kubernetes-cluster-test-1

#This should return an output similar to this one
Request to create cluster 27cdcad8-375f-4d4f-a186-8fa99b80c5c5 accepted
#This indicates that the command was successful and the cluster is being built

```

A cluster containing one master node and one worker node takes approximately 14 minutes to build. By default, cluster creation times out at 60 minutes.

After the cluster has been created, you can associate a floating IP to the master node and SSH into the node using:

```
` ssh -i <mykeypair-private-key> fedora@<floating_ip> `
```

## References

<https://docs.openstack.org/magnum/train/user/#>

<https://clouddocs.web.cern.ch/containers/quickstart.html>

## 18.3 Submitting jobs to a Kubernetes Cluster

**Warning:** Deprecated

Magnum is deprecated and will be replaced in the future with Cluster API.

A Kubernetes job creates one or more pods on a cluster and have the added benefit of being retried until a specified number of pods successfully terminate. Jobs are described by YAML and can be executed using *kubectl*.

### 18.3.1 Submitting jobs

Jobs are defined by a YAML config with a *kind* parameter of Job. Below is an example job config for computing  $\pi$  to 2000 places.

Listing 1: job.yaml

```
apiVersion: batch/v1
kind: Job
metadata:
  name: pi
spec:
  template:
    spec:
      containers:
      - name: pi
        #Docker image
        image: perl
        #Compute pi to 2000 places by running this command in the container
        command: ["perl", "-Mbignum=bpi", "-wle", "print bpi(2000)"]
      restartPolicy: Never
      #Number of retries to attempt before stopping (default is 6)
      backoffLimit: 4
```

To run this job use:

```
kubect1 apply -f ./job.yaml
```

This will result in the creation of a pod with a single container named *pi* that is based on the *perl* image. The specified command, equivalent to `perl -Mbignum=bpi -wle "print bpi(2000)"`, will then be executed in the container. You can check on the status of the job using

```
kubect1 describe jobs/pi
```

Then the output from the container can be obtained through

```
pods=$(kubect1 get pods --selector=job-name=pi --output=jsonpath='{.items[*].metadata.name}')
kubect1 logs $pods
```

**Important:** The job and pods it creates are usually kept after execution to allow the logs to be seen, these can be deleted using `kubect1 delete jobs/pi` or `kubect1 delete -f ./job.yaml`.

One way to avoid a build up of undeleted jobs and pods is to use *ttlSecondsAfterFinished* which was declared stable as of Kubernetes v1.23. This defines the time after which a job in a Complete/Failed state should be deleted automatically. For example the following job will be deleted 100 seconds after finishing.

```
apiVersion: batch/v1
kind: Job
metadata:
  name: pi
spec:
  #Delete the job 100 seconds after successful completion
  ttlSecondsAfterFinished: 100
```

(continues on next page)

(continued from previous page)

```

template:
  spec:
    containers:
    - name: pi
      image: perl
      command: ["perl", "-Mbignum=bpi", "-wle", "print bpi(2000)"]
    restartPolicy: Never

```

### 18.3.2 Parallel execution

The above example runs a single pod until completion or 4 successive failures. It is also possible to execute multiple instances of pods in parallel. For a simple example we can require a *fixed completion count* by assigning `.spec.completions` in the YAML file to require more than one successful execution of a pod is required before the job is considered complete. We can then also specify `.spec.parallelism` to increase the number of pods that can be running at any one time. For example, the below will run up to 2 pods in parallel until 8 of them finish successfully.

```

apiVersion: batch/v1
kind: Job
metadata:
  name: pi
spec:
  ttlSecondsAfterFinished: 100
  #Require 8 successful completions before the job is considered finished
  completions: 8
  #Allow up to 2 pods to be running in parallel
  parallelism: 2
  template:
    spec:
      containers:
      - name: pi
        image: perl
        command: ["perl", "-Mbignum=bpi", "-wle", "print bpi(2000)"]
      restartPolicy: Never

```

If one pod fails a new pod will be created to take its place and the job will continue.

You can also use a *work queue* for parallel jobs by not specifying `.spec.completions` at all. In this case the pods should coordinate amongst themselves or via an external service to determine when they have finished as when any one of them successfully exits the job will be considered complete. Therefore each should exit only there is no more work for *any* of the pods.

### 18.3.3 Scheduling jobs

To run jobs on a schedule you can use CronJobs. These are also described using a YAML file, for example:

Listing 2: cronjob.yaml

```
apiVersion: batch/v1
kind: CronJob
metadata:
  name: hello
spec:
  #This described when the job described below should be executed
  schedule: "* * * * *"
  jobTemplate:
    spec:
      template:
        spec:
          containers:
            - name: hello
              image: busybox:1.28
              imagePullPolicy: IfNotPresent
              command: ["/bin/sh", "-c", "date; echo Hello from the Kubernetes cluster"]
          restartPolicy: OnFailure
```

This will run a job every minute that prints “Hello from the Kubernetes cluster”. You can create the CronJob using:

```
kubectl create -f ./cronjob.yaml
```

You may check on the status of the CronJob using `kubectl get cronjob hello` and watch the jobs it creates in real time using `kubectl get jobs --watch`. From the latter you will see that the job names appear as *hello-* followed by some numbers e.g. *hello-27474266*. This can be used to view the output of the job using

```
pods=$(kubectl get pods --selector=job-name=hello-27474266 --output=jsonpath='{.items[*].
  ↪metadata.name}')
kubectl logs $pods
```

The *schedule* parameter, which in this case causes the job to run every minute, is in the following format.

```
minute (0 - 59), hour (0 - 23), day of the month(1 - 31), month (1 - 12), day of the
  ↪week (0 - 6)
```

Where the day of the week is 0 for Sunday and 6 for Saturday. In the example the asterisk is used to indicate *any*. You may find tools such as <https://crontab.guru/> helpful in writing these. For example `5 4 * * 2` will run at 04:05 on Tuesdays. The time specified is the local time of the machine.

You may delete the CronJob, along with any of its existing jobs and pods using

```
kubectl delete cronjob hello
```

### 18.3.4 References

<https://kubernetes.io/docs/concepts/workloads/controllers/job/>

<https://kubernetes.io/docs/tasks/job/automated-tasks-with-cron-jobs/>

## 18.4 JupyterHub on Kubernetes

**Warning:** Deprecated

Magnum is deprecated and will be replaced in the future with Cluster API.

This documentation assumes that you will be installing JupyterHub on a Kubernetes cluster that has been created using OpenStack Magnum.

**In this tutorial we will break the installation down into the following:**

- Create a cluster template and launch a Kubernetes cluster using OpenStack Magnum
- Install Helm v3 and define persistent volume for the cluster
- Install JupyterHub

### 18.4.1 Creating a Kubernetes Cluster

The template for the cluster:

```
openstack coe cluster template create --coe kubernetes \
    --image cf37f7d0-1d6b-4aab-a23b-df58542c59cb \
    --external-network External \
    --network-driver flannel \
    --volume-driver cinder \
    --dns-nameserver 8.8.8.8 \
    --flavor c1.medium \
    --master-flavor c1.medium \
    --docker-volume-size 10 \
    --docker-storage-driver devicemapper \
    --labels kube_tag=v1.14.3,kube_dashboard_enabled=1,
↪heat_container_agent_tag=train-stable-3,auto_healing=true,ingress_controller=traefik
    --server_type vm
    test-template
```

Create a cluster:

```
openstack coe cluster create --cluster-template test-template \
    --keypair mykeypair \
    --docker-volume-size 10 \
    --master-count 1 \
    --node-count 1 \
    test-cluster
```

*#This should return an output similar to this one*

(continues on next page)

(continued from previous page)

```
Request to create cluster 27cdcad8-375f-4d4f-a186-8fa99b80c5c5 accepted
#This indicates that the command was successful and the cluster is being built
```

Once the cluster has been created successfully, we can associate a floating IP to the master node VM and then SSH into the cluster:

```
ssh -i mykeypair.key fedroa@FLOATING_IP

#This should return something similar to:

Last login: Fri Sep 18 13:17:02 2020 from 130.XXX.XXX.XXX

[fedora@test-template-vbo5u2doyiao-master-0 ~]$

#You have now successfully connected to the master node
```

## Configure Storage

Magnum does not automatically configure cinder storage for clusters.

The storage class can be defined using a YAML file. For example we could define the storage class to be:

**YAML**            **File**            **from:**            [https://github.com/zonca/jupyterhub-deploy-kubernetes-jetstream/blob/master/kubernetes\\_magnum/storageclass.yaml](https://github.com/zonca/jupyterhub-deploy-kubernetes-jetstream/blob/master/kubernetes_magnum/storageclass.yaml)

```
apiVersion: storage.k8s.io/v1

kind: StorageClass

metadata:

  name: standard

  annotations:

    storageclass.beta.kubernetes.io/is-default-class: "true"

  labels:

    kubernetes.io/cluster-service: "true"

    addonmanager.kubernetes.io/mode: EnsureExists

provisioner: kubernetes.io/cinder
```

Then we create the storage class:

```
kubectl create -f storageclass.yaml
```

### 18.4.2 Helm v3

The Train release supports Helm v2 charts being installed and supports labels for installing Tiller.

However, it is possible to install and run charts for Helm v3.

In the Ussuri release onwards, Magnum supports the use of a label to install Helm v3 client. This label can be added to a template or at cluster creation time.

**Note:** Helm v2 reaches end of support in November 2020

To install Helm 3:

```
curl https://raw.githubusercontent.com/helm/helm/master/scripts/get-helm-3 | bash
```

Other methods for installing Helm v3 can be found here: <https://helm.sh/docs/intro/install/>

Now Helm v3 has been installed, we can install JupyterHub.

### 18.4.3 JupyterHub

The following is the tutorial from the [\\_Zero to JupyterHub with Kubernetes\\_](#) installation documentation.

```
# Generate a random hex string
openssl rand -hex 32 #copy the output
```

Then create a file called config.yaml and write the following:

```
vi config.yaml # fedora doesn't use nano
```

```
proxy:
  secretToken: "<RANDOM_HEX>" #this is the random string which you have copied
```

Next is to add the JupyterHub Helm chart to your chart repository and install it.

```
helm repo add jupyterhub https://jupyterhub.github.io/helm-chart/
helm repo update

RELEASE=jhub
NAMESPACE=jhub

helm upgrade --cleanup-on-fail \
  --install $RELEASE jupyterhub/jupyterhub \
  --namespace $NAMESPACE \
  --create-namespace \
  --version=0.9.0 \
  --values config.yaml \
  --timeout 30m0s #This is to stop the installation from timing out
```

When installation is complete it should return a message similar to the following:

```
NAME: jhub
LAST DEPLOYED: Tue Oct 13 11:01:15 2020
NAMESPACE: jhub
STATUS: deployed
```

(continues on next page)

(continued from previous page)

REVISION: 1

TEST SUITE: None

NOTES:

Thank you for installing JupyterHub!

Your release is named jhub and installed into the namespace jhub.

You can find if the hub and proxy is ready by doing:

```
kubectl --namespace=jhub get pod
```

and watching for both those pods to be in status 'Running'.

You can find the public IP of the JupyterHub by doing:

```
kubectl --namespace=jhub get svc proxy-public
```

It might take a few minutes for it to appear!

Note that this is still an alpha release! If you have questions, feel free to

1. Read the guide at <https://z2jh.jupyter.org>
2. Chat with us at <https://gitter.im/jupyterhub/jupyterhub>
3. File issues at <https://github.com/jupyterhub/zero-to-jupyterhub-k8s/issues>

## 18.4.4 References:

<https://github.com/zonca/jupyterhub-deploy-kubernetes-jetstream>

<https://zonca.dev/2020/05/kubernetes-jupyterhub-jetstream-magnum.html>

<https://zero-to-jupyterhub.readthedocs.io/en/latest/>

<https://helm.sh/docs/intro/install/>

## 18.5 Autoscaling Clusters

**Warning:** Deprecated

Magnum is deprecated and will be replaced in the future with Cluster API.

The Cluster Autoscaler (CA) is a feature in OpenStack Magnum that can be enabled in order for the cluster to scale up or down the worker nodegroup. The default version which the Train release uses is v1.0. The version of CA to use can be changed at cluster creation by using the label `autoscaler_tag`

This feature can be enabled by using the label `auto_scaling_enabled=true` in a cluster template or at cluster creation.

**Note:** The CA will start attempting to scale down nodes as long as the machine IDs for each node matches their VM ID.

### 18.5.1 Machine IDs

On nodes in a Kubernetes cluster, the system UUID matches the ID of the VM hosting that node. However, the Cluster Autoscaler uses the machine ID to refer to the node when the cluster needs to be scaled down and a node removed. Kubernetes reads the ID for the VMs from the file `/etc/machine-id` on each VM in the cluster. However, these IDs may not match the IDs of the VMs. If the machine ID and system UUID (VM ID) on a node do not match, then the following errors may be present in the CA pod's log:

```
E1215 12:59:58.084833      1 scale_down.go:932] Problem with empty node deletion:
↳failed to delete cluster-node-5: manager error deleting nodes: could not find stack
↳indices for nodes to be deleted: 1 nodes could not be resolved to stack indices
E1215 12:59:58.084958      1 static_autoscaler.go:430] Failed to scale down: failed to
↳delete at least one empty node: failed to delete cluster-node-5: manager error
↳deleting nodes: could not find stack indices for nodes to be deleted: 1 nodes could
↳not be resolved to stack indices
I1215 13:10:09.265757      1 scale_down.go:882] Scale-down: removing empty node cluster-
↳node-5
I1215 13:10:18.362187      1 magnum_manager_heat.go:347] Could not resolve node
↳{Name:cluster-node-5 MachineID:d6580d63b98346daacd54c644f76bbd6 ProviderID:openstack://
↳/d07e9c8f-e7dd-4342-9ba6-f5c912afc04e IPs:[10.0.0.8]} to a stack index
```

To update the machine ID to match the VM ID, the file can be edited directly using:

```
vi /etc/machine-id
```

*#Then replace the machine ID with the VM's ID*

After a few minutes Kubernetes will have updated the IDs for the nodes on those VMs. The system UUID and the machine ID can be seen using `kubectl describe node <node-name>`.

For example:

```
$ kubectl describe node cluster-node-3
Name:                cluster-node-3
Roles:               <none>
Labels:              beta.kubernetes.io/arch=amd64
                    beta.kubernetes.io/instance-type=8
                    beta.kubernetes.io/os=linux
                    draino-enabled=true
                    failure-domain.beta.kubernetes.io/region=RegionOne
                    failure-domain.beta.kubernetes.io/zone=ceph
                    kubernetes.io/arch=amd64
                    kubernetes.io/hostname=cluster-node-3
                    kubernetes.io/os=linux
                    magnum.openstack.org/nodegroup=default-worker
                    magnum.openstack.org/role=worker
Annotations:         flannel.alpha.coreos.com/backend-data: {"VtepMAC":"7e:d2:52:53:d3:8c
↳"}
                    flannel.alpha.coreos.com/backend-type: vxlan
                    flannel.alpha.coreos.com/kube-subnet-manager: true
                    flannel.alpha.coreos.com/public-ip: 10.0.0.131
                    node.alpha.kubernetes.io/ttl: 0
                    volumes.kubernetes.io/controller-managed-attach-detach: true
CreationTimestamp:   Mon, 16 Nov 2020 15:45:32 +0000
```

(continues on next page)

(continued from previous page)

```

Taints:                <none>
Unschedulable:         false
Conditions:
  Type                  Status  LastHeartbeatTime          LastTransitionTime
  ↳ Reason              ↳ Message
  ---
  ↳ -----
  KernelDeadlock        False   Tue, 15 Dec 2020 13:06:50 +0000  Fri, 27 Nov 2020 11:45:15 +0000
  ↳ KernelHasNoDeadlock kernel has no deadlock
  ReadonlyFilesystem    False   Tue, 15 Dec 2020 13:06:50 +0000  Fri, 27 Nov 2020 11:45:15 +0000
  ↳ FilesystemIsNotReadOnly
  MemoryPressure         False   Tue, 15 Dec 2020 13:07:32 +0000  Mon, 16 Nov 2020 15:45:32 +0000
  ↳ KubeletHasSufficientMemory kubelet has sufficient memory available
  DiskPressure           False   Tue, 15 Dec 2020 13:07:32 +0000  Mon, 16 Nov 2020 15:45:32 +0000
  ↳ KubeletHasNoDiskPressure kubelet has no disk pressure
  PIDPressure            False   Tue, 15 Dec 2020 13:07:32 +0000  Mon, 16 Nov 2020 15:45:32 +0000
  ↳ KubeletHasSufficientPID kubelet has sufficient PID available
  Ready                  True    Tue, 15 Dec 2020 13:07:32 +0000  Fri, 27 Nov 2020 11:44:41 +0000
  ↳ KubeletReady         kubelet is posting ready status
Addresses:
  InternalIP: 10.0.0.131
Capacity:
  cpu: 2
  ephemeral-storage: 39922Mi
  hugepages-1Gi: 0
  hugepages-2Mi: 0
  memory: 4030348Ki
  pods: 110
Allocatable:
  cpu: 2
  ephemeral-storage: 37675125903
  hugepages-1Gi: 0
  hugepages-2Mi: 0
  memory: 3927948Ki
  pods: 110
System Info:
  Machine ID: 54d6093e-0b15-4c92-80f7-5f3126e06083
  System UUID: 54d6093e-0b15-4c92-80f7-5f3126e06083
  Boot ID: dddb9b89-559c-4f3f-8b1f-6b6f0d5a62dd
  ↳ ...

```

This shows that this node had the machine ID updated so that it now matches the System UUID and will refer to the VM by the correct ID if the Cluster AutoScaler attempts to remove the node when scaling the cluster.

The Cluster Autoscaler will begin to successfully scale down nodes once machine IDs match VM IDs. To prevent a node being scaled down, the following annotation needs to be added to the node:

```

kubectl annotate node <node-name> cluster-autoscaler.kubernetes.io/scale-down-
↳ disabled=true

```

This will indicate to CA that this node cannot be removed from the cluster when scaling down.

## 18.5.2 Cluster Autoscaler Deployment

The deployment of the CA on the cluster will be similar to the following:

```
$ kubectl describe deployment cluster-autoscaler -n kube-system
Name:                cluster-autoscaler
Namespace:           kube-system
CreationTimestamp:    Mon, 16 Nov 2020 12:55:53 +0000
Labels:              app=cluster-autoscaler
Annotations:         deployment.kubernetes.io/revision: 1
                    kubect1.kubernetes.io/last-applied-configuration:
                    {"apiVersion":"apps/v1","kind":"Deployment","metadata":{"
→ "annotations":{},"labels":{"app":"cluster-autoscaler"},"name":"cluster-autoscaler"...
Selector:            app=cluster-autoscaler
Replicas:            1 desired | 1 updated | 1 total | 1 available | 0 unavailable
StrategyType:        RollingUpdate
MinReadySeconds:     0
RollingUpdateStrategy: 25% max unavailable, 25% max surge
Pod Template:
  Labels:             app=cluster-autoscaler
  Service Account:    cluster-autoscaler-account
  Containers:
    cluster-autoscaler:
      Image:          docker.io/openstackmagnum/cluster-autoscaler:v1.0
      Port:           <none>
      Host Port:      <none>
      Command:
        ./cluster-autoscaler
        --alsologtostderr
        --cloud-provider=magnum
        --cluster-name=686f9fa1-eb56-4c23-9afd-67a79c283736
        --cloud-config=/config/cloud-config
        --nodes=1:4:default-worker
        --scale-down-unneeded-time=10m
        --scale-down-delay-after-failure=3m
        --scale-down-delay-after-add=10m
      Environment:    <none>
      Mounts:
        /config from cloud-config (ro)
        /etc/kubernetes from ca-bundle (ro)
  Volumes:
    ca-bundle:
      Type:          Secret (a volume populated by a Secret)
      SecretName:    ca-bundle
      Optional:      false
    cloud-config:
      Type:          Secret (a volume populated by a Secret)
      SecretName:    cluster-autoscaler-cloud-config
      Optional:      false
  Conditions:
    Type            Status    Reason
    ----            -
    Progressing     True      NewReplicaSetAvailable
```

(continues on next page)

(continued from previous page)

```

Available      True      MinimumReplicasAvailable
OldReplicaSets: cluster-autoscaler-8669c48d54 (1/1 replicas created)
NewReplicaSet:  <none>
Events:         <none>

```

We can see in the Command can change the time the autoscaler waits before determining that a node is unneeded and should be scaled down. We can also change the delay time between adding nodes during scaling up and the amount of time to wait after scaling down fails.

### 18.5.3 Example: A Cluster Scaling Up

Let's have a cluster that has CA enabled and consists of one master node and one node. If the worker node is cordoned and nginx pods still need to be scheduled, the CA will send an OpenStack request to resize the cluster and increase the node count from 1 to 2 in order to have a node available to schedule a node. This can be seen in the container or pod logs for the CA:

```

2020-11-16T11:00:13.721916753Z I1116 11:00:13.721164 1 scale_up.go:689] Scale-up:
↪setting group default-worker size to 2
2020-11-16T11:00:21.441786855Z I1116 11:00:21.441504 1 magnum_nodegroup.go:101]
↪Increasing size by 1, 1->2
2020-11-16T11:00:59.763966729Z I1116 11:00:59.763422 1 magnum_nodegroup.go:67]
↪Waited for cluster UPDATE_IN_PROGRESS status

```

You should see the stack for the cluster being updated on OpenStack and see the node visible in the cluster:

```

ssh -i <mykey.pem> fedora@<master-node-ip>

kubectl get nodes

NAME                                STATUS    ROLES    AGE    VERSION
cluster-test-master-0              Ready     master   2d20h   v1.14.3
cluster-test-node-1                Ready,SchedulingDisabled  <none>   2d20h   v1.14.3
cluster-test-node-2                NotReady  <none>   0s      v1.14.3

#Here we can see that the new node has been spun up and it being set up

#After the node has been configured it reports that it is ready

kubectl get nodes

NAME                                STATUS    ROLES    AGE    VERSION
cluster-test-master-0              Ready     master   2d20h   v1.14.3
cluster-test-node-1                Ready,SchedulingDisabled  <none>   2d20h   v1.14.3
cluster-test-node-2                Ready     <none>   0s      v1.14.3

```

## 18.5.4 References

<https://docs.openstack.org/magnum/train/user/>

<https://github.com/kubernetes/autoscaler/tree/master/cluster-autoscaler/cloudprovider/magnum>

## 18.6 OpenStack Magnum Users

**Warning:** Deprecated

Magnum is deprecated and will be replaced in the future with Cluster API.

When a cluster is created, Magnum creates unique credentials for each cluster. This allows the cluster to make changes to its structure (e.g. create load balancers for specific services, create and attach cinder volumes, update the stack, etc.) without exposing the user's cloud credentials.

### 18.6.1 How to find the Magnum User Credentials

We can obtain the cluster credentials directly from the VM which the master node is on. First, SSH into the master node's VM and then:

```
[fedora@cluster-master-0 ~]$ cd /etc
[fedora@cluster-master-0 /etc]$ cd kubernetes/
[fedora@cluster-master-0 kubernetes]$ ls
#This will return the list of items similar to:

apiserver      cloud-config    controller-manager  kube_openstack_config
↪manifests          scheduler
ca-bundle.crt  cloud-config-ocm  get_require_kubeconfig.sh  kubelet
↪proxy
certs          config          keystone_webhook_config.yaml  kubelet-config.yaml
↪proxy-kubeconfig.yaml

#The cluster's credentials can be found in the file 'cloud-config'
#Print the config to the terminal
[fedora@cluster-master-0 kubernetes]$ cat cloud-config
```

This will return the cloud-config file containing the cluster's credentials similar to:

```
[Global]
auth-url=https://AUTH-URL
user-id=CLUSTER_USER_ID
password=PASSWORD
trust-id=TRUST-ID
ca-file=/etc/kubernetes/ca-bundle.crt
region=RegionOne
[LoadBalancer] #with the octavia ingress controller enabled
use-octavia=True
subnet-id=SUBNET_ID
floating-network-id=FLOATING_NETWORK_ID
```

(continues on next page)

(continued from previous page)

```
create-monitor=yes
monitor-delay=1m
monitor-timeout=30s
monitor-max-retries=3
[BlockStorage]
bs-version=v2
```

These global variables should be used when setting up configmaps such as magnum-auto-healer configmap. **Never** use your own cloud credentials in Kubernetes configmaps. These would be visible to anyone who has access to the master node in the cluster.

## LEGACY DOCUMENTATION

### 19.1 JupyterHub Handover Documentation

#### 19.1.1 Overview

This project has been a proof of concept of how the STFC cloud team can provide a JupyterHub service for existing cloud users. Since Jupyter services are popular among the research community and are an easy way of setting up a Python development environment to develop aspects of code, it was decided it would be a good addition to the services offered, with the hope of making use of the STFC cloud.

The proof of concept instance uses LDAP authentication so users can login using the federal ID and password. When logged in, users can create one or multiple notebook servers. This makes use of the STFC cloud where a notebook server is hosted on one VM, giving good scalability.

#### 19.1.2 Launching Test Instance

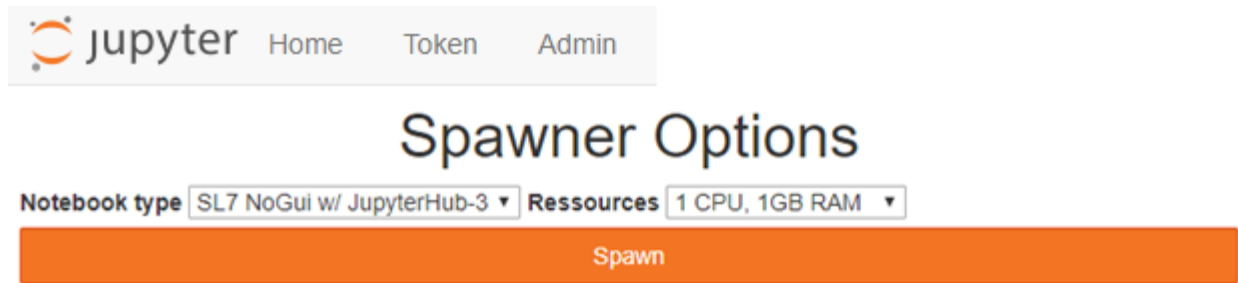
To launch the proof of concept, SSH into `host-XXX-XXX-XXX-XXX.nubes.stfc.ac.uk` and execute the following commands:

```
1 cd /home/USERNAME
2 source ssl_env
3 source jupyterhub/venv3.6/bin/activate
4 cd jupyterhub/test-hub
5 jupyterhub
```

#### 19.1.3 Using JupyterHub - User's Perspective

The majority of the JupyterHub user experience is documented here: <https://jupyterhub.readthedocs.io/en/stable/reference/urls.html>. However, some specific notes for this instance:

Once JupyterHub has been started, users can start to access it, `XXX.XXX.XXX.XXX:8000` as per the current config. By default, this will send the user to `hub/spawn`, which gives options to create a new notebook server:



Clicking on the ‘Home’ button (top left of the page) will allow the user to see a list of their notebook servers. This will send the user to `hub/home` - I think this should be the user’s ‘homepage’.

Start My Server

## Named Servers

In addition to your default server, you may have additional server(s) with names. This allows you to have more than one server running at the same time.

Server name	URL	Last activity	Actions
<input type="text" value="Name your server"/> <a href="#">Add New Server</a>			
lpd-tile-testing-server	/user/ /lpd-tile-testing-server	25 minutes ago	<a href="#">stop</a>
test-ip		a day ago	<a href="#">start</a> <a href="#">delete</a>
test-name		4 days ago	<a href="#">start</a> <a href="#">delete</a>
test-server		a day ago	<a href="#">start</a> <a href="#">delete</a>
test2		4 days ago	<a href="#">start</a> <a href="#">delete</a>
test3		4 days ago	<a href="#">start</a> <a href="#">delete</a>

This homepage will show a list of the user’s **named servers**. Users can start, stop and delete these servers. To create a new named server, just type the name into the text field and click ‘Add New Server’.

There are two ways of making a notebook server in JupyterHub. You can create a named server as described above, or create a server using the ‘Start My Server’ button. Prior to named server functionality, this was the only way for users to create a notebook server. Servers created in this way aren’t part of the named servers and as such, won’t appear in that list. Using this method of server creation means you can only have one notebook server of this type. You can still use named servers of course. Essentially, why use this old functionality if you can make use of the named servers?

Sidenote, if the user cannot make use of the ‘multiple servers per user’ functionality, check the JupyterHub version. This wasn’t supported in UI format until version 1.0.0.

### 19.1.4 Setting SSL Environment Variable

In order to correctly setup the authentication, I needed to set an SSL key passphrase. Before the JupyterHub instance is started, an environment variable needs to be set so JupyterHub can make use of the passphrase. This is done in a small script called `ssl_env`. A small snippet of logging is shown below which highlights what to look for when the environment variable hasn’t been set before launching JupyterHub.

```
_tls_common.js:104
  c.context.setKey(options.key, options.passphrase);
[...]
```

RuntimeError: Proxy failed to start with **exit** code 1

### 19.1.5 Recreating Instance

JupyterHub requires Python 3 so I used Python 3.6 throughout for this proof of concept.

Inside `/home/USERNAME/jupyterhub/test-hub`, there's a `requirements.txt` file which is the output of a `pip freeze` done on the virtual environment used for this proof of concept. If you don't have access to this, I installed: `jupyterhub`, `jupyterhub-ldapauthenticator` and `libcloudspawner` using `Pip`.

To launch notebook servers on OpenStack, JupyterHub must be installed on the VM. The easiest way to do this is by creating a VM using an image that has it pre-installed. You will need to create an image which has JupyterHub (including Python 3.6) installed, along with any software required by your community and a 'user folder' - this is the intended base folder for notebook servers (this option is set in the `jupyterhub` config file). As of writing, this folder is `/home/USERNAME/jupyterhub-user-folder`.

This JupyterHub instance I used has a host certificate with it. This is because authentication uses SSL. A self signed certificate will work for testing or various tiers of more trusted certificates can be requested via [cloud-support@stfc.ac.uk](mailto:cloud-support@stfc.ac.uk)

It should be noted the project needs a security group to allow port 8000.

### 19.1.6 LDAP Authentication

For JupyterHub to function, a spawner and an authenticator is needed. These can be custom made, or a community made solution can be used. For the proof of concept, I've made use of community solutions.

The default authentication method is to use Unix accounts local to the host. Obviously, this isn't a good solution for a service as big as the STFC cloud. The configuration for this authenticator has been based off the LDAP config for the STFC cloud web interface, but needed to be modified to work for this purpose.

There are two LDAP authenticators for JupyterHub: `jupyterhub-ldapauthenticator` (used in the proof of concept) and `jupyterhub-ldap-authenticator`. The latter is based on the prior, with support for multiple LDAP servers, among other things (full list of features shown here: <https://pypi.org/project/jupyterhub-ldap-authenticator/>). I decided to use the initial library due to better documentation (and existing users) and there appeared to be no need to have the additional features.

Beyond this demo, there could be plans to add further authentication choices. The next method to look at would be the IRIS IAM service. This uses OAuth authentication, which is the technology used by GitHub and Google to allow users to sign into other websites, using credentials from the respective services.

- GitHub repo of the LDAP authenticator: <https://github.com/jupyterhub/ldapauthenticator>
- An OAuth authenticator has already been written: <https://github.com/jupyterhub/oauthenticator>
- Information about the IRIS IAM service: <https://iris-iam.stfc.ac.uk/privacypolicy/>

### 19.1.7 LibCloudSpawner

A good introduction to spawners is written on JupyterHub's documentation: <https://jupyterhub.readthedocs.io/en/stable/reference/spawners.html>

As touched upon in the overview, this spawner allows scalability of the service. Instead of putting strain of the user's demands on the host VM, this allows a single VM per notebook server. Currently, there's only functionality to create these VMs in a single project. I use the Packer service account to create VMs, simply because this is an account I had easy access to at the time. A separate service account should really be made for this work. The VMs are created in a specific project because this is where the JupyterHub host VM is placed and for proof of concept, it wasn't worthwhile to spend the time moving the VM outside of this project.

I put my key on the VMs (`ex_keyname` in the config) so I can get into them if needed. If this service went further, giving the cloud team the ability to access users VMs in case of something going wrong would be vital.

When VMs are created, it's not exceptionally clear which VMs link to which notebook servers. I got confused on the small scale of 6/7 VMs I had up at a time, so an improvement needs to be made to prevent confusion on a larger scale. Even just a mapping between VM name/UUID and notebook server name (formatted as 'username/notebook-server-name') would help a great deal. A limit of number of servers per user can be added, though isn't currently configured.

I've been communicating (via Gitter) with the developer of this spawner to get assistance on how to set it up and get things working. I've also been submitting issues to the GitHub repo which he responds to and pushes out updates fairly quickly. This could be useful in the future to add new functionality.

- GitHub repo: <https://github.com/tristanlt/jupyter-libcloudspawner>
- ReadTheDocs: <https://jupyter-libcloudspawner.readthedocs.io/en/latest/index.html>
- Gitter Conversations: <https://gitter.im/jupyter-libcloudspawner/community>

### 19.1.8 Example Notebook Server - LPD Tile Testing

To ensure this instance works, I got an existing Jupyter notebook to function with JupyterHub. This notebook takes data files from a detector, and tests the tiles/chips on the detector function as they should.



As per the screenshot, the notebook functions as expected and displays a number of data visuals. This notebook requires certain Python libraries to function. In order to make it work, I SSH'd into the VM and installed them via Pip. If the cloud team don't want end users accessing those VMs via SSH, users can create terminals within JupyterHub. However, some thought might need to be given if users aren't allowed to access certain parts of the VMs. At the very least, management of images/allowing users to have a Python environment that'll work for them will be important.

- GitHub repo of the notebook from the state I used: <https://github.com/stfc-aeg/lpd-tile-testing/tree/b3378da7c86e3810523485343e4642de0ec3ca28>

### 19.1.9 Config Notes

When first starting JupyterHub, a config file is generated for you. I've used this file and appended to it as required. Anything that's *important* has been un-commented and made use of, but to summarise:

```
c.JupyterHub.cleanup_proxy = False
c.JupyterHub.cleanup_servers = False
```

When the JupyterHub service is shutdown, this means notebook servers won't be deleted. In the case of the LibCloudSpawner, VMs won't be changed at all (deleted, for example).

```
c.JupyterHub.ssl_cert = 'file.crt'
c.JupyterHub.ssl_key = 'file.key'
```

The certificates from e-Science come in a .pem format. I converted this to a crt and key to fit with JupyterHub's configuration.

```
c.Spawner.start_timeout = 360
```

6 minutes is enough time to ensure OpenStack has done its job and beyond that, something has definitely gone wrong. The timeout used to be 90 seconds and this wasn't always enough time for the spawner to get an IP address from the VM it created.



## INDICES AND TABLES

- `genindex`
- `modindex`
- `search`