
Chameleon Cloud Documentation

Release 2.0a

Chameleon Cloud

Mar 21, 2024

INTRODUCTION

1	Welcome to Chameleon	1
2	Getting started	3
3	Sign in with federated identity	11
4	PI eligibility	17
5	Project management	19
6	User profile	29
7	Getting help	31
8	Graphical User Interface (GUI)	33
9	Command Line Interface (CLI)	49
10	Jupyter interface	57
11	Overview	63
12	Resource discovery	65
13	Reservations	75
14	Bare metal instances	91
15	Images	105
16	Monitoring	113
17	Complex appliances	121
18	Object Store	145
19	Shares	155
20	Networking	163
21	FPGAs	181
22	Experiment Precs	185

23 Trovi sharing portal	195
24 Daypass	201
25 KVM	205

WELCOME TO CHAMELEON

1.1 What is Chameleon?

Chameleon is an NSF-funded testbed system for Computer Science experimentation. It is designed to be deeply reconfigurable, with a wide variety of capabilities for researching systems, networking, distributed and cluster computing and security. Chameleon's features include:

- Bare metal access to hardware, via the cloud
- A wide variety of hardware types, including:
 - Infiniband
 - NVMe
 - GPUs/FPGAs
 - Low-power Xeon and ARM processors
- A powerful set of networking capabilities, including:
 - Isolated layer-2 networks
 - SDN/OpenFlow
 - Dedicated WAN layer-2 links

1.2 Using Chameleon

Chameleon is available to Computer Science researchers and students. To access the system, follow the instructions in *Getting started*. Find out more at <https://www.chameleoncloud.org>.

GETTING STARTED

This guide will walk you through the initial steps of getting an account, joining a project and working with your first instance.

- *Step 1: Log in to Chameleon*
- *Step 2: Create or join a project*
- *Step 3: Start using Chameleon!*
 - *The Chameleon dashboard*
 - *Reserving a node*
 - *Launching an instance*
 - *Associating an IP address*
 - *Accessing Your Instance*

2.1 Step 1: Log in to Chameleon

Just click the “Log in” button situated in the top right corner of [our main page](#) – you probably won’t even need to create an account!

If your institution is a member of [InCommon](#) (most US research and education institutions are) – or if you have a Google account – you can log in with your institutional/Google credentials via the federated login. Otherwise, the log in process will guide you to create an account ([read more about logging into Chameleon via federated login](#)).

On your first Chameleon login you will be asked to accept [terms and conditions](#) of use. Please, note that as part of those terms and conditions you are requested to acknowledge Chameleon in publications produced using the testbed: see our FAQ for information on [how to reference Chameleon in your publications](#) and the suggested [acknowledgement text](#).

Once you log in, you will be able to [edit your Chameleon profile](#), sign up for webinars, and participate in our community. However, to actually use the testbed you will first need to **join or create a project** (see below).

2.2 Step 2: Create or join a project

To get access to Chameleon resources, you will need to be associated with a **project** that is assigned a **resource allocation**.

If you want to **join an existing Chameleon project**, you will need to ask the PI of the project to add your username. You can find your username in [your Chameleon profile](#)—it is also displayed in the top-right corner when you are logged in.

If you want to **create a project**, you will first either need to obtain a PI status or work with somebody who has PI status. To determine if you can obtain PI status, please see a [list of PI eligibility criteria](#). If you do not meet these criteria (e.g., students generally do not), you will need to ask your advisor or other scientist supervising your research to create the project for you. You can request PI status by checking a box in [your Chameleon profile](#). Chameleon PI status requests are typically reviewed within one business day.

Once you have PI status, you may apply for a new project with an initial allocation. A project application typically consists of a short description of your intended research and takes one business day to process. Once your project has been approved, you will be able to utilize the testbed sites.

2.3 Step 3: Start using Chameleon!

Congratulations, you are now ready to launch your first instance! Instances are much like what you may expect to find in a virtual machine, except here the instances are on bare metal nodes - the core feature of Chameleon. A bare metal node is a whole physical server that you have exclusive access to. An instance is going to be a bare metal node that has been launched with an operating system image. Follow these steps to make a reservation for a node, launch an instance and log in to it.

Note: Chameleon also offers a multi-tenant, virtualized cloud, with fewer functionalities and a smaller scale. See [KVM](#) for more details.

2.3.1 The Chameleon dashboard

Chameleon resources are available at multiple sites, e.g., [CHI@TACC](#) and [CHI@UC](#). When you access one of the sites, you are first taken to a dashboard, which shows a summary of your project's current resource usage and get quick access to each of the sites. The dashboard looks something like this:

2.3.2 Reserving a node

First, we need to reserve a node for our use. Chameleon provides bare metal access to nodes. When you create a reservation for one or more nodes, only you and other users on your project will be able to use those nodes for the time specified. We will create a single day reservation for a compute node, which are the most common types of nodes available on Chameleon.

1. In the sidebar, click *Reservations*, then click *Leases*
2. Click on the + *Create Lease* button in the toolbar
3. Type *my_first_lease* for the lease name
4. Find the *Resource Properties* section. In the dropdown below *node_type*, select *compute_skylake*
5. Click the *Create* button

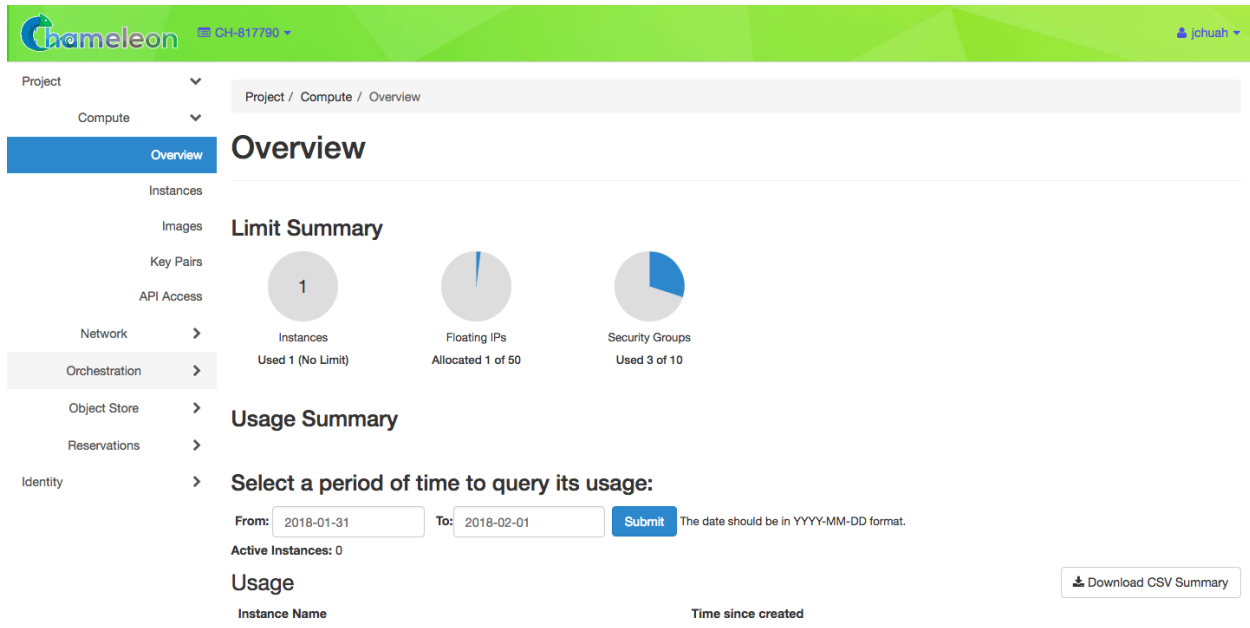


Fig. 1: An overview of your project’s current resource usage

The reservation will start shortly, at which point you can launch an instance on a bare metal node.

Note: You have created an “on demand” reservation. When you do not specify a start date or time in the future, the reservation will start as soon as possible and will last one day.

Important: Do not attempt to stack reservations to circumvent the 7-day lease limitation. Your leases may be deleted. Please refer to our [best practices](#) if you require a longer reservation.

2.3.3 Launching an instance

Once the reservation starts, you can launch a bare metal instance on the node that has been leased to you.

1. In the sidebar, click *Compute*, then click *Instances*
2. Click on the *Launch Instance* button in the toolbar and the *Launch Instance* wizard will load
3. Type *my_first_instance* for the instance name and select your *my_first_lease* reservation
4. Click *Source* in sidebar. Then, find *CC-CentOS8* in the image list and click the *Up* arrow to select it.
5. Click *Keypair* in sidebar. Click the *+ Create Key Pair* button and enter *mychameleonkey* for the key name. This will automatically start a download for a file named *mychameleonkey.pem*. This is your private key pair that you will use to access your instance.
6. Click the *Launch Instance* button.

Congratulations, you have launched an instance on a bare metal node!

Create Lease

Lease Name *
my_first_lease

Start Date ?
Today

Start Time ?
Now

Reservation Length in Days ?
1

End Date ?
Tomorrow

End Time ?
Same time as now

Minimum Number of Hosts ?
1

Maximum Number of Hosts ?
1

Resource Properties ?

- node_type = compute_haswell X

Add Filter

Description:

Create a new lease with the provided values.

Leave date and time values blank to start a lease immediately (on-demand).

For specific node reservations, you can find the node UUID using [Resource Discovery](#) on the user portal.

Cancel

Create

Fig. 2: The Create Lease dialog - be sure to select compute_skylake in the dropdown below node_type

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 *

Reservation *

Count *

Total Instances (No Limit)

1 Current Usage

1 Added

✕ Cancel

< Back

Next >

Launch Instance

Fig. 3: Enter an instance name and select your reservation

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

Allocated

Name	Updated	Size	Type	Visibility
> CC-CentOS7	1/2/18 10:59 AM	816.30 MB	qcow2	Public

▼ Available 83

Q Click here for filters.

Select one

Name	Updated	Size	Type	Visibility
> image-ubuntu-trusty-jenkins-cc-trusty-builder-7	1/5/18 6:52 PM	502.49 MB	qcow2	Public
> CC-Ubuntu16.04-CUDA8	1/5/18 6:33 PM	2.22 GB	qcow2	Public
> CC-Ubuntu16.04	1/3/18 12:35 AM	668.48 MB	qcow2	Public

Fig. 4: Select the CC-CentOS8 image

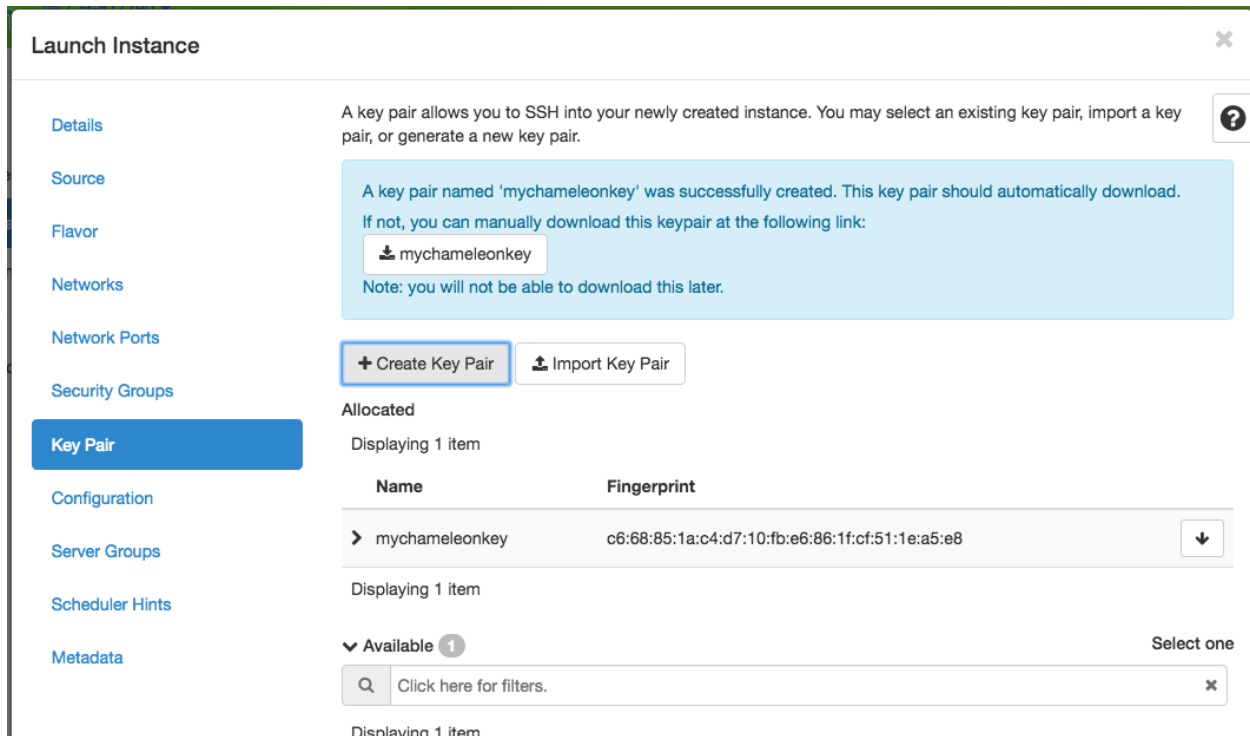
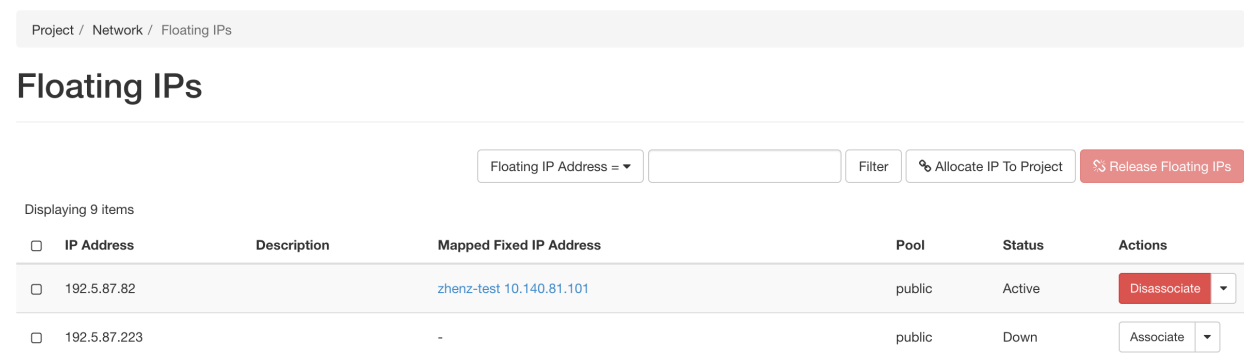


Fig. 5: You can create or import a public/private keypair for accessing your instance.

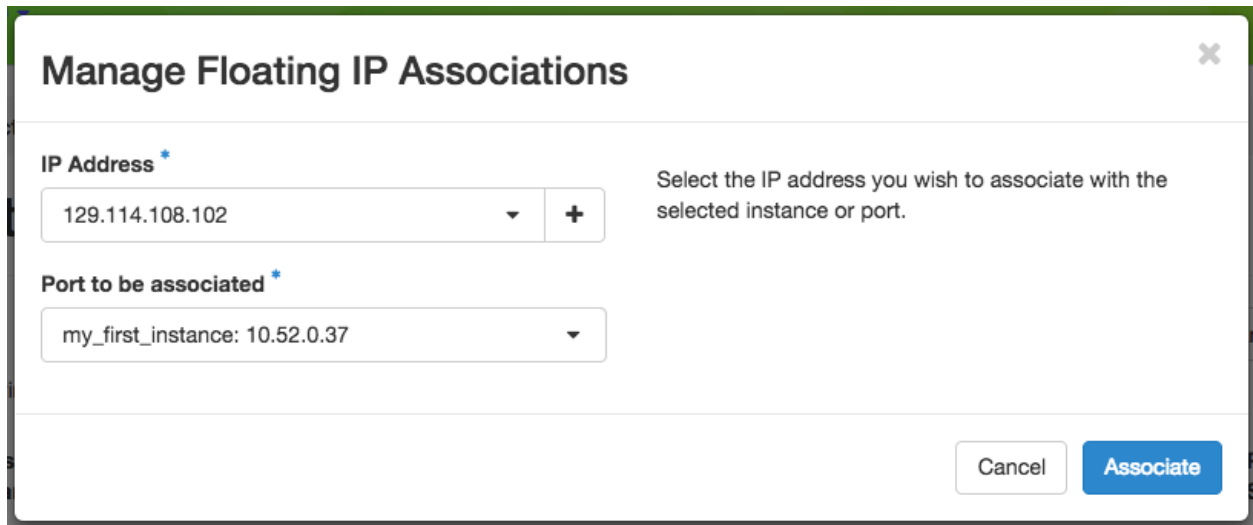
2.3.4 Associating an IP address

Your instance may take approximately ten minutes to launch. The launch process includes powering up, loading the operating system over the network, and booting up for the first time on a rack located either at the University of Chicago or the Texas Advanced Computing Center, depending on where you chose to launch your instance. Before you can access your instance, you need to first assign a floating IP address - an IP address that is accessible over the public Internet.

1. Go to the *Floating IP* dashboard by clicking on *Network* and *Floating IPs* in the sidebar.



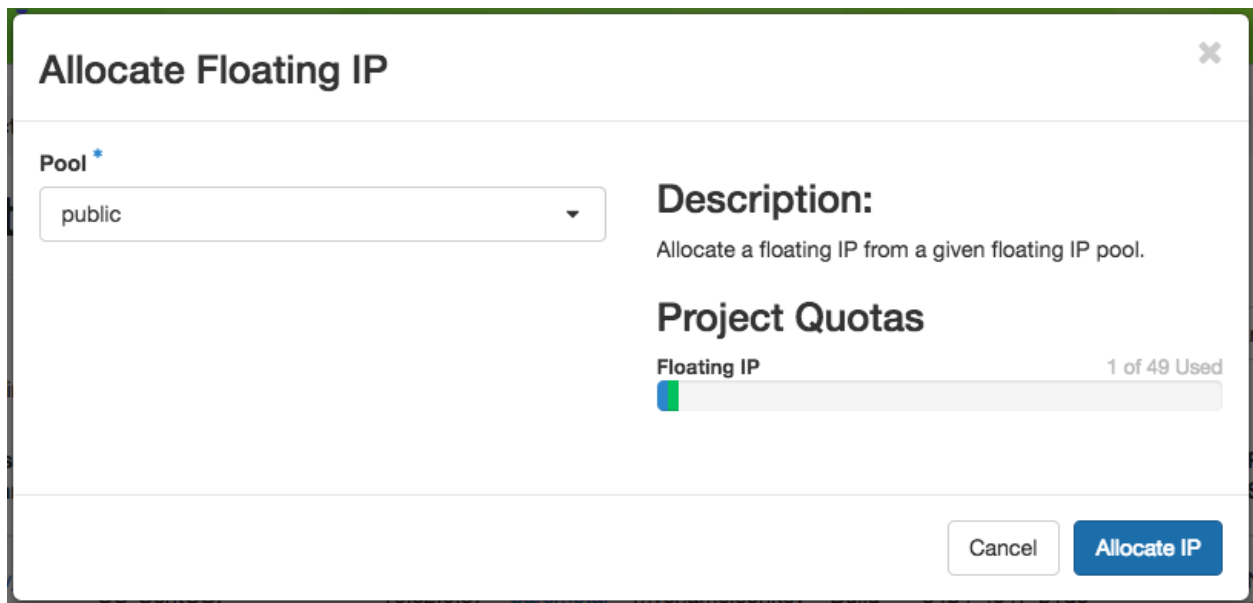
2. If you have a Floating IP not currently associated to an instance, click the *Associate* button for the IP. A dialog will load that allows you to assign a publicly accessible IP to your instance. Click the *Associate* button in the dialog to complete the process of associating the public IP to your instance.
3. If you didn't already have a Floating IP available, you may allocate one to your project by clicking on the *Allocate*



The dialog titled "Manage Floating IP Associations" features a close button (X) in the top right corner. It contains two main input sections. The first section, labeled "IP Address" with a blue asterisk, includes a dropdown menu showing "129.114.108.102" and a plus icon to the right. The second section, labeled "Port to be associated" with a blue asterisk, includes a dropdown menu showing "my_first_instance: 10.52.0.37". To the right of these inputs is a text instruction: "Select the IP address you wish to associate with the selected instance or port." At the bottom right, there are two buttons: "Cancel" and "Associate".

Fig. 6: Here you can assign a floating IP address

IP to Project button along the top row in the Floating IP dashboard. A new dialog will open for allocating the floating IP.



The dialog titled "Allocate Floating IP" features a close button (X) in the top right corner. It contains a "Pool" dropdown menu with "public" selected. To the right of this is a "Description:" section with the text "Allocate a floating IP from a given floating IP pool." Below the description is a "Project Quotas" section. It includes a "Floating IP" label and a progress bar showing "1 of 49 Used". At the bottom right, there are two buttons: "Cancel" and "Allocate IP".

Fig. 7: This dialog allows you to allocate an IP address from Chameleon's public IP pool

Click the *Allocate IP* button. The Floating IP dashboard will reload and you should see your new Floating IP appear in the list. You can now go back to step 2.

2.3.5 Accessing Your Instance

Once your instance has launched with an associated floating IP address, it can be accessed via SSH using the private key that you downloaded during the *Launching an Instance* step.

Note: The following instructions assume that you are using a macOS or Linux terminal equivalent. You may view our [YouTube video on how to login via SSH on Windows](#).

To log in to your instance, follow these steps:

1. Open a terminal window and navigate to where you downloaded the `mychameleonkey.pem` file. Change the permissions on the file to user read/write only:

```
chmod 600 mychameleonkey.pem
```

2. Add the key to your current SSH identity:

```
ssh-add mychameleonkey.pem
```

3. Log in to your Chameleon instance via SSH using the `cc` user account and your floating IP address. If your floating IP address was `129.114.108.102`, you would use the command:

```
ssh cc@129.114.108.102
```

Note: Change the IP address in this command to match your instance's floating IP address!

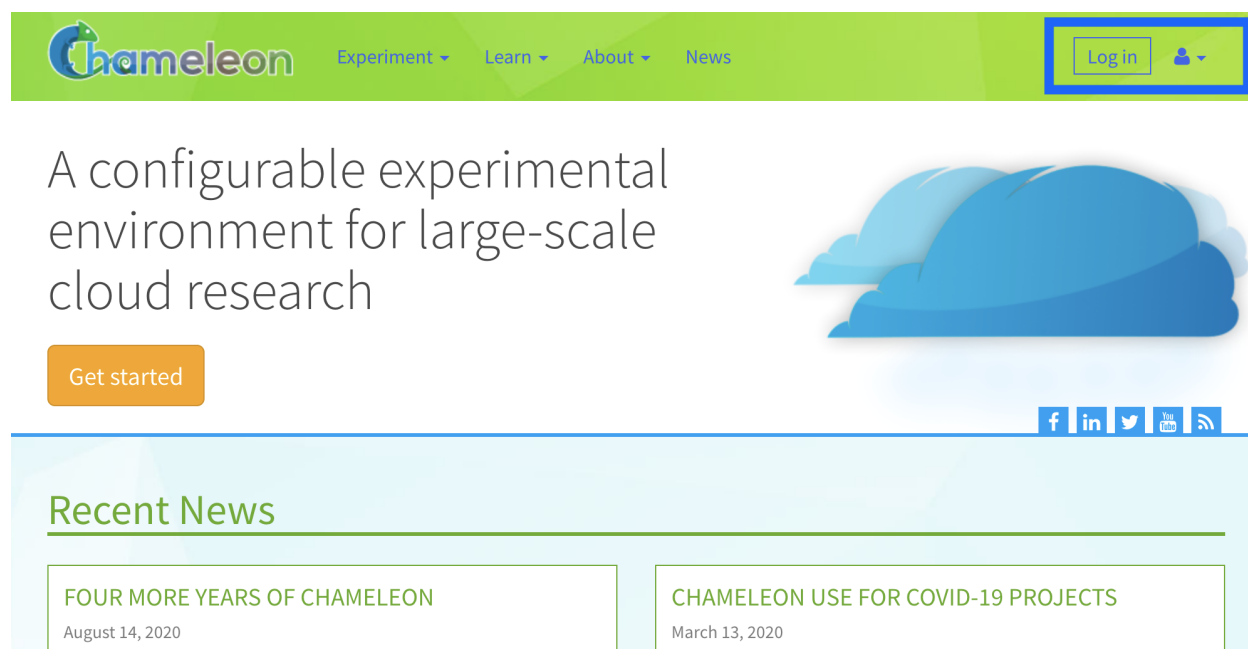
SIGN IN WITH FEDERATED IDENTITY

Federated login enables users to use a single set of credentials to log into many different services. For example, federated login allows you to use your university or other institutional credentials to log into Chameleon—there is no need to create a new account. In addition, since federated login is supported by many testbeds and services across scientific infrastructures you will be able to sign in once and use multiple services.

Chameleon uses [Globus Auth](#), a popular authentication service, to implement federated login and federates with entities supported by Globus. Users can sign in using their existing institutional account if their institution is an [InCommon](#) member, use their Google account, or create a [Globus ID](#) tied to an email and password that they provide. In addition, Chameleon also federates with the TAS entity.

3.1 Logging in

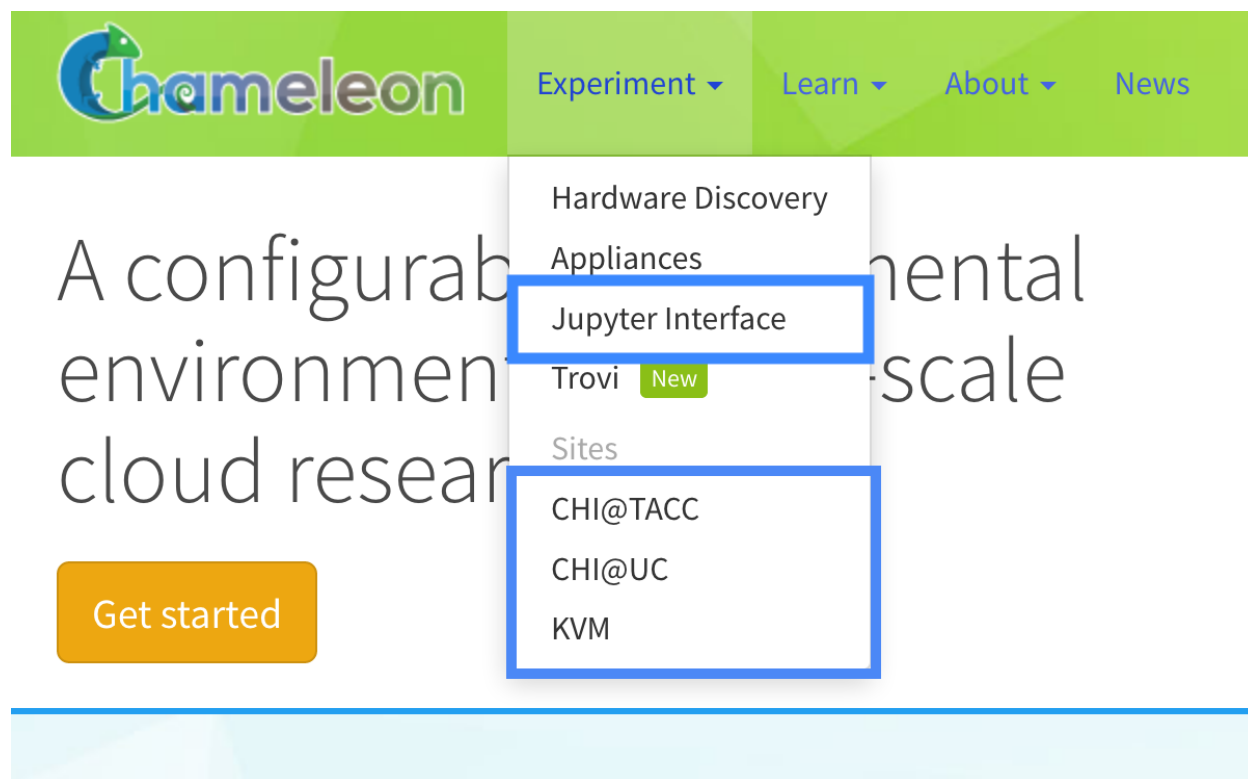
To log in to the Chameleon user portal, where you can manage your projects, user profile, and submit [Help Desk](#) tickets, use the “Log in” button.



To log in to any of the testbed sites ([CHI@TACC](#), [CHI@UC](#), [KVM@TACC](#)) or the *Jupyter environment*, just click their item in the “Experiment” dropdown on [chameleoncloud.org](#). The login process is triggered automatically.

Important: You must be part of project with an active allocation to use the testbed sites! Refer to our “*getting started*” *guide* for more info.

Note: You can bookmark the URLs to the testbed sites and Jupyter environment if you want to access them directly in the future.



You will be taken to a Single Sign On (SSO) page with a few options for how to authenticate. The options are:

- **Sign in via federated identity:** this option allows you to re-use your existing institution, research lab, or university credentials to log in to Chameleon. It requires your host institution to participate in the [InCommon](#) federation.
- **Google:** this option allows you to sign in with any Google account.
- **ORCID:** you can also sign in with a valid ORCID account.
- **TAS:** sign in via the TAS entity.
- For the time being, if you are an existing user, you can also leave SSO and go back to the old sign in page, where you sign in with your Chameleon username and password.

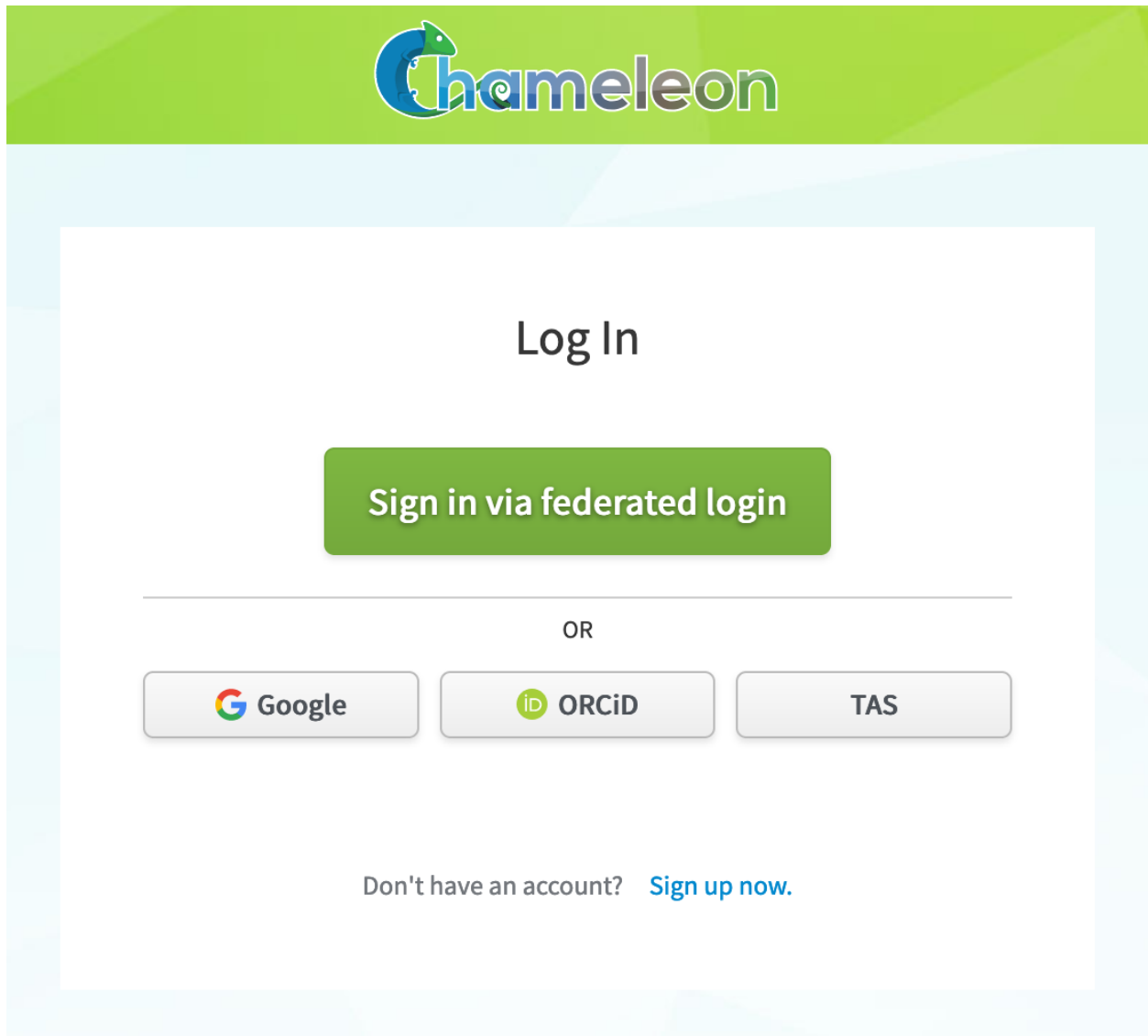


Fig. 1: The Single Sign On (SSO) portal login page.

3.2 Legacy users (created before November 2020)

Users who signed up for a Chameleon account prior to November 2020 must migrate their account to use federated identity. Refer to the following step-by-step guide for more details.

3.2.1 Migrating to federated identity

Federated login enables users to use a single set of credentials to log into many different services. For example, federated login allows you to use your institutional credentials to log into Chameleon—there is no need to create a new account. To leverage its [many benefits](#), Chameleon is migrating to federated identity and will eventually drop support for signing in via legacy Chameleon accounts. Existing Chameleon users can link their existing Chameleon account to a federated identity account so that they preserve their project memberships, disk images, volumes, SSH key pairs, and other data saved over time.

To ease the transition, during the [migration period](#) you can log in via your old Chameleon account as well as the federated account. Existing users however are highly encouraged to migrate to a federated account as soon as possible. Read more about federated identity, how it has been implemented, as well as the migration schedule in [our FAQ](#).

Important: Existing Chameleon users should go through the following account linking process **before** attempting to sign in with federated identity. Signing in before linking accounts can lead to you accidentally creating a new Chameleon account, if we cannot match the email address on your federated account to the one you used when registering for Chameleon.

Step-by-step instructions

1. **Sign in** to the [Chameleon identity management interface](#). You will see a screen that prompts you for your existing Chameleon username and password.
2. Once authenticated, you will be taken to a page showing all the identities (credentials) linked to your Chameleon user account. **Click the “Add” button** next to the entry for the Globus identity to initiate another login via federated identity. You can pick whichever login method you choose; your federated identity need not be tied to the same email address or username as your existing Chameleon account if you wish.

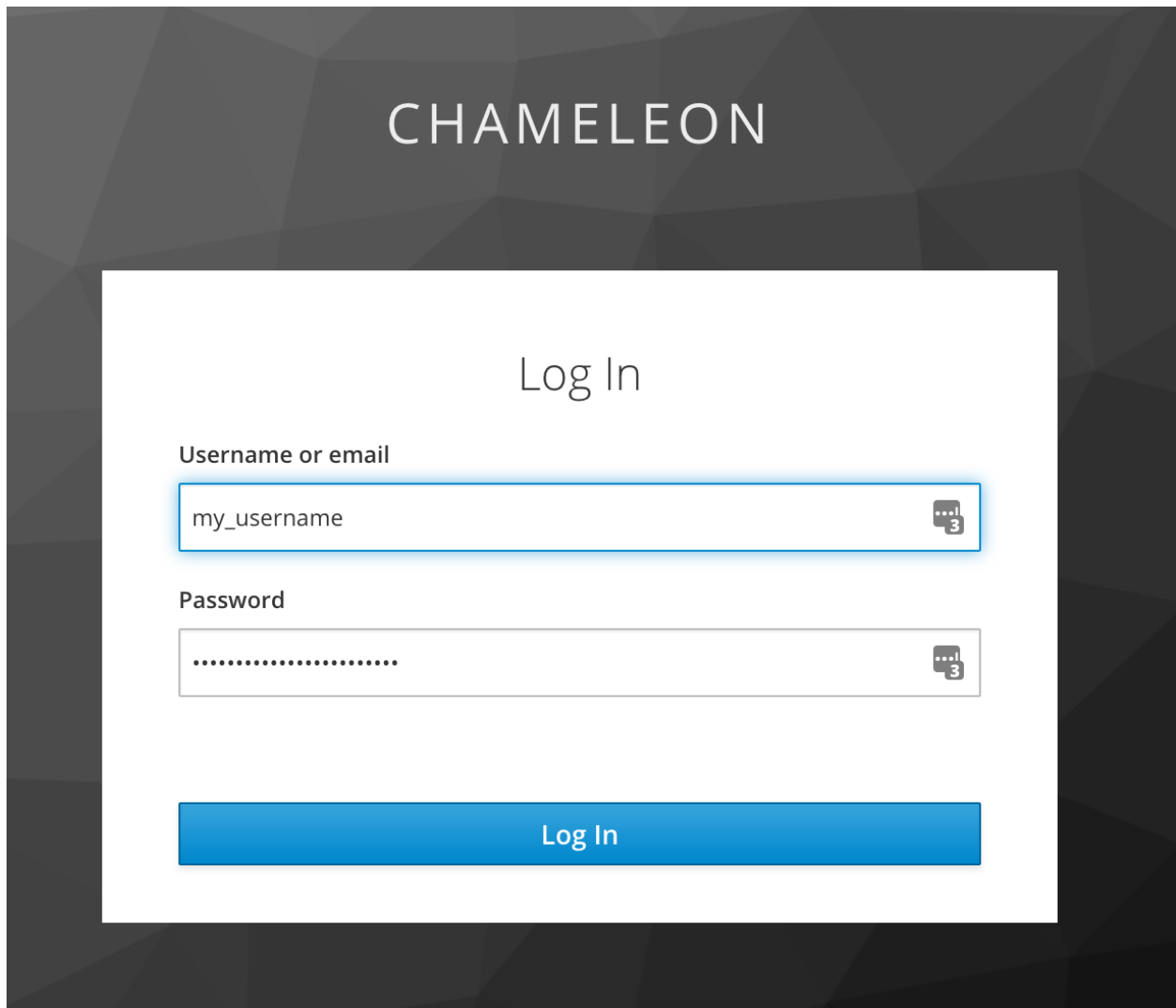
Once added, your linked federated identity can be used to log in to your existing Chameleon account in the future, and you no longer need to use your Chameleon username and password to login to any part of the Chameleon infrastructure.

3. If you try to log in to any of the testbed sites with your linked federated identity, you may notice that some of your data is missing, namely:
 - *disk images, snapshots* and *volumes* you or other project members have created in the past,
 - *active or pending leases* you made,
 - *SSH keypairs* you previously associated with your account, and
 - *active server instances* launched recently

Migrating to federated identity effectively creates a new account for you in the system, so in order to retain access to some of your saved data, a short data migration is necessary.

Go to the [migration page](#) in the Chameleon user portal to start this process (**note:** ensure you now log in to the user portal using federated login, and not the old sign-in page!)

You should see a page that looks like the following.



The image shows the Chameleon Cloud login interface. At the top, the word "CHAMELEON" is displayed in a large, white, sans-serif font against a dark, geometric background. Below this, a white rectangular box contains the login form. The form is titled "Log In" in a medium-sized font. It features two input fields: "Username or email" and "Password". The "Username or email" field contains the text "my_username" and has a small icon of a speech bubble with the number "3" next to it. The "Password" field is filled with dots and also has a similar icon. Below the input fields is a large blue button labeled "Log In".

Federated Identities

Globus Auth

TACC

Add

Migrate account

Your user
jason_a Migrate

Projects

CH-820687 Re-run migration
Migrated on 10/7/2020, 5:02:31 PM

Chameleon Ops Migrate

CH-820156 Re-run migration
Migrated on 10/8/2020, 12:15:20 PM

chameleon_sab Re-run migration
Migrated on 10/7/2020, 5:01:17 PM

Select a user or project to start (or re-run) a migration.

A **user migration** ensures any SSH keypairs uploaded in the past are transferred over to your new account.

A **project migration** ensures any disk images or snapshots associated with the project are transferred to the new account. Disk images and snapshots will still be accessible via the old account, so this is non-destructive. Any member of the project can initiate a migration.

Important: Not all data can be migrated to your new account. Leases and active server instances will remain on your old account. If you wish to access these, you can log in to the testbed site using the old sign in method.

4. **Trigger a migration of your user** to copy over any SSH keypairs you previously associated with your account.
5. **Trigger migrations of projects** you are a member of to associate any disk images, snapshots and volumes to your new account. This only needs to be done once by any member of the project, but can also be re-run in the event that you or other project members create disk images/snapshots under their old account.
6. You should now have access to your old data via your new account linked to your federated identity!

Using the CLI

If you use the *command line interface* when interacting with Chameleon (or use another tool that interfaces directly with Chameleon's APIs), you will no longer be able to authenticate with a username and password. You should re-download your *RC file* and use it when invoking the CLI, as it will have new authentication parameters compatible with your account pre-filled.

You can also look in to generating an *application credential* for your command line client or app, which may be simpler.

PI ELIGIBILITY

Chameleon PIs carry significant responsibility for the users on their projects; we therefore limit PI eligibility to individuals from the following groups:

- **Academic institutions:** This eligibility criterion covers research scientists, research staff, and faculty members in supervisory positions at academic institutions. Graduate and PhD student researchers (including those serving as paid research assistants) are **not** typically considered eligible for PI status on Chameleon. Students should instead ask their faculty advisor to request PI status and give them access to a project.
- **Federal agencies such as national labs, R&D centers, and institutes:** Research staff employed by federal agencies or non-NSF Federally Funded R&D Centers (FFRDCs) are eligible to apply for an allocation.
- **International research institutions:** To promote intellectual exchange and federation with institutions abroad we support a limited number of international PIs with ongoing, active collaborations with scientists in the US.
- **NSF Graduate Student Fellows:** While in most cases, a graduate student is ineligible to be PI of an allocation request, an exception is made for NSF Graduate Student Fellows. Recipients of these NSF awards can submit requests for Startup allocations as long as they include supporting documentation (grant number or an award letter) as part of the request submission.
- **Independent museums, observatories, libraries, research laboratories, professional societies, and similar organizations** in the United States that are directly associated with educational or research activities are eligible.
- **State educational offices or organizations and local school districts** may submit allocation requests intended to broaden the impact, accelerate the pace, and increase the effectiveness of improvements in science, mathematics, and engineering education in both K-12 and post-secondary levels. A teacher or educator at an accredited public or private K-12 school is eligible to apply for an allocation as PI.

We do occasionally provide case-by-case exceptions to this guideline in well-justified cases.

PROJECT MANAGEMENT

Project management tasks, such as adding users to your project or requesting a renewal, is performed through the portal at <https://chameleoncloud.org>. After you have [registered](#) and verified your email address, you may [login to the portal](#). Once logged in, you should be at *Dashboard* page automatically. If not, you can access your *Dashboard* via the dropdown list on top right of the screen.

5.1 Dashboard

The Dashboard's main page consists of two control panels - the *Active Projects* control panel and the *Open Tickets* panel.

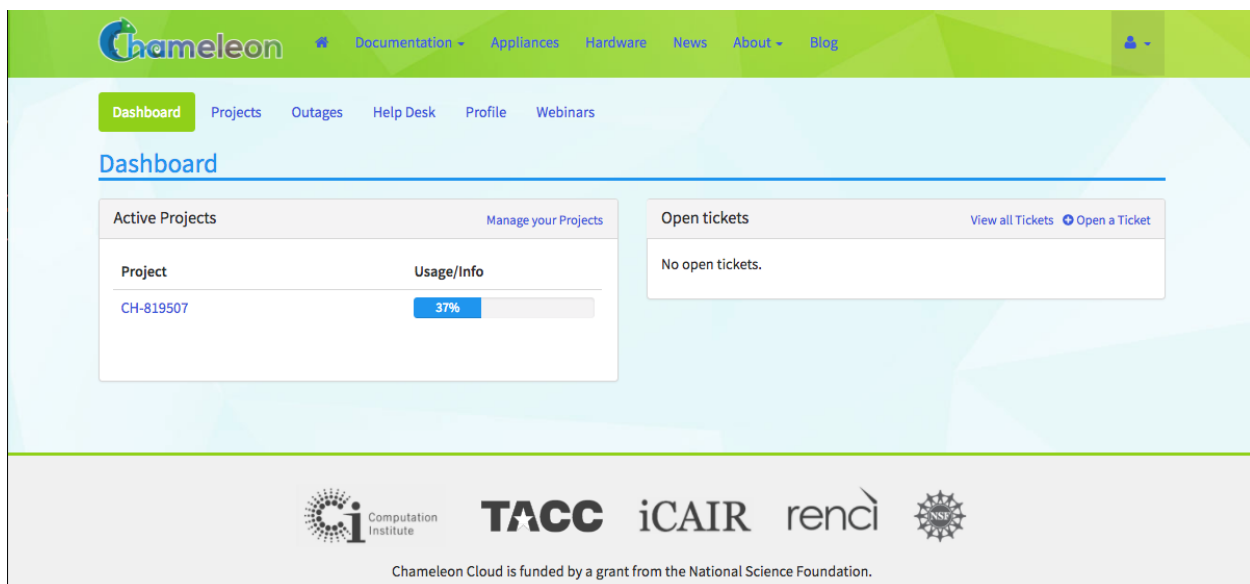


Fig. 1: The project dashboard

The *Active Projects* control panel allows you to view all your active projects and their current usage. You may click on a project to view details.

The *Open Tickets* panel lists all your active help desk tickets. In addition, you can [Open a Ticket](#) via the *Open Tickets* panel.

5.2 Projects

The Dashboard's [Projects Page](#) allows you to manage your current projects.

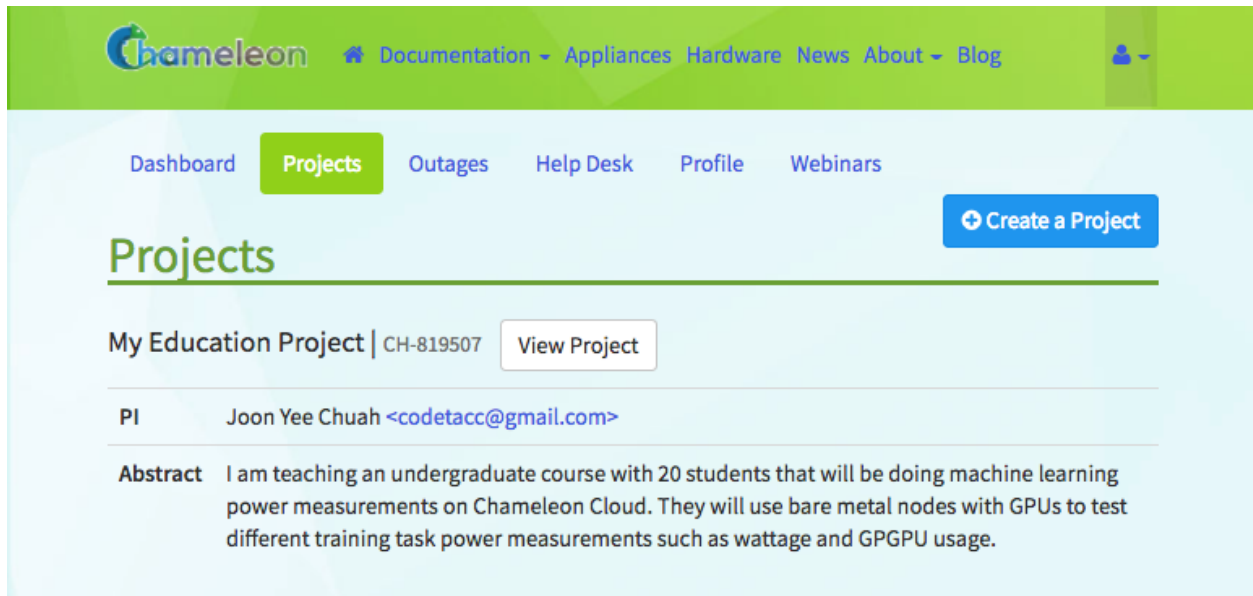


Fig. 2: Project list

Each individual *Project* has its own:

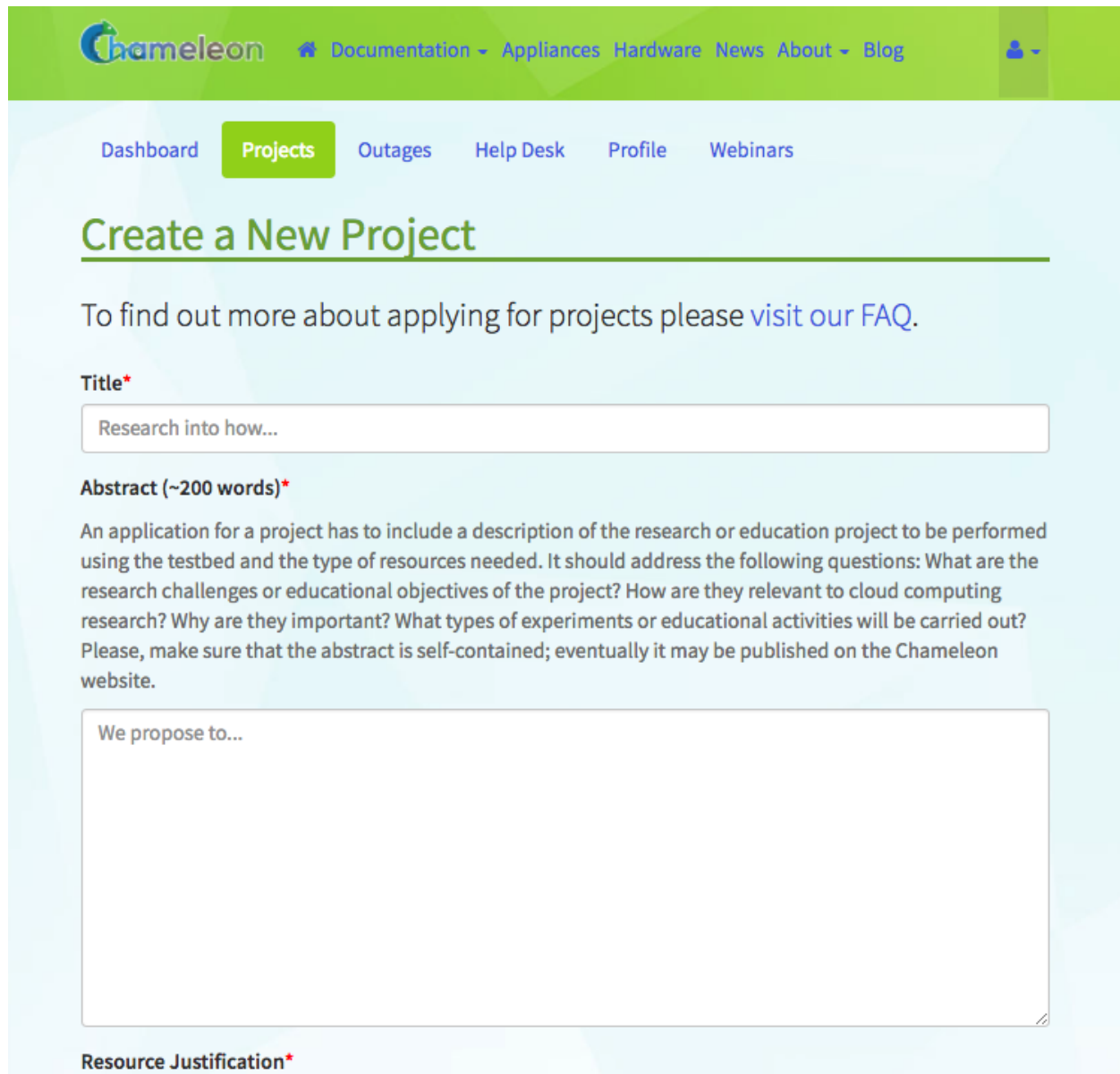
- Service Unit allocations
- Users that have access to the project
- System resources such as *Security Groups*, *Floating IP Addresses* and *Instances*
- Assets such as snapshots, object containers, metrics and network configuration

5.2.1 Creating a Project

To create a project, click the [+Create a Project](#) button. After filling out and submit the request form, a system administrator will review your request and notify you once your project get approved. Project durations are six months with a default allocation of 20,000 *Service Units*.

5.2.2 Service Units

One Service Unit (SU) is equivalent to one hour of usage of one allocatable resource (physical hosts, network segments, or floating IPs). For example, a reservation for 5 Skylake compute nodes for 8 hours would use 40 SUs. However, for certain types of resources, more SUs will be charged. For more details about allocation charges, please see [here](#).



The screenshot shows the 'Create a New Project' form in the Chameleon Cloud interface. The top navigation bar is green with the Chameleon logo and links to Documentation, Appliances, Hardware, News, About, and Blog. Below this, a secondary navigation bar has links to Dashboard, Projects (highlighted), Outages, Help Desk, Profile, and Webinars. The main heading is 'Create a New Project'. Below the heading, there is a link to 'visit our FAQ'. The form consists of three main sections: 'Title*' with a text input field containing 'Research into how...'; 'Abstract (~200 words)*' with a paragraph of instructions and a large text area containing 'We propose to...'; and 'Resource Justification*' which is partially visible at the bottom.

Create a New Project

To find out more about applying for projects please [visit our FAQ](#).

Title*

Research into how...

Abstract (~200 words)*

An application for a project has to include a description of the research or education project to be performed using the testbed and the type of resources needed. It should address the following questions: What are the research challenges or educational objectives of the project? How are they relevant to cloud computing research? Why are they important? What types of experiments or educational activities will be carried out? Please, make sure that the abstract is self-contained; eventually it may be published on the Chameleon website.

We propose to...

Resource Justification*

Fig. 3: The Create a New Project form

5.2.3 Project Details

Clicking on a project from either the *Dashboard* main page or the *Projects* page will allow you to manage one of your approved *Projects*.

The screenshot shows the 'Project Details' page for a project named 'MPICH appliance on Chameleon' with ID 'CH-818644'. The page has a navigation bar with links: Dashboard, Projects, Outages, Help Desk, Profile, Webinars, and Publications. A blue button 'Add Publications' is in the top right.

The project details section includes:

- PI:** Zhuo Zhen <zhenz@uchicago.edu> (with an external link icon)
- Nickname:** mpich (with an external link icon)
- Abstract:** The purpose of this project is to create and maintain an MPICH appliance on Chameleon, in collaboration with the MPICH software maintainers at Argonne National Laboratory.
- Tag:** Other — My project research area doesn't fit in any of the predefined categories (with an external link icon)

Below the details is the 'Allocations' section, which states 'No current allocations' and provides a 'Toggle inactive allocations' link. A message explains that no allocations are shown because the project is new or the allocation has expired, and suggests clicking 'Request Allocation'.

A green '+ Request Allocation' button is present. Below it is a table with headers: Status, Date Requested, Date Reviewed, Start On, End On, and Usage.

The 'Project Members' section includes a form to 'Add a User to Project' with a text input for 'Username or email' and buttons for 'Add user', 'Add multiple users', and 'Remove multiple users'. There is also a 'Set Default SU Budget for All Members' section with a text input set to '0' and a 'Set Default Budget' button.

Two user entries are shown in a table:

projectmanager@uchicago.edu		Project, Manager		Remove user	
Email: anishreddy@uchicago.edu	Role: Manager	Submit	Cancel	Used SU: 0	SU Budget: 0

projectmember@uchicago.edu		Project, Member		Remove user	
Email: markpowers@uchicago.edu	Role: Member	Submit	Cancel	Used SU: 0	SU Budget: 0
				<input type="range"/>	Set

Fig. 4: Project details

In the details page of your project, you may *recharge or extend your allocation*, *view allocation usage details*, and *manage users* of your project.

5.2.4 Recharge or Extend Your Allocation

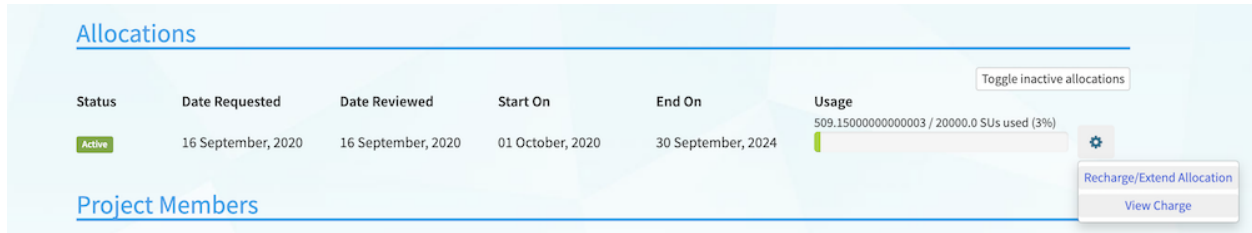


Fig. 5: Allocation actions

In the *Allocations* section of your *Project Details*, you may view your project start and end dates, current *Service Unit* usage and request a *Service Unit* recharge or project extension. To request a *Service Unit* recharge or *Project* extension, click the *gear* button at the end of the allocation row, and then click *Recharge/Extend Allocation*. When requesting renewal or recharge of the allocations, we may ask you to update your *publications dashboard*, so keeping it up to date now can save you time later!

5.2.5 View Allocation Usage Details

Name	Email	Region	Resource ID	Resource Type	Start On	End On	Used SUs
		CHI@TACC	1d5ff928-9e20-4f9d-80ff-761876843a5c	physical:host	Oct. 26, 2020, 5:47 p.m.	Oct. 28, 2020, 5:46 p.m.	95.97
		CHI@TACC	ff86775e-45cc-424d-92ac-ce037ff96906	physical:host	Oct. 27, 2020, 8:12 a.m.	Oct. 27, 2020, 6 p.m.	19.6
		CHI@UC	909d8b6c-18c3-47be-a1c1-e51fa00f5fb0	physical:host	Oct. 13, 2020, 4:43 p.m.	Oct. 13, 2020, 4:44 p.m.	0.03
		CHI@UC	9856d211-4caf-4051-a969-7a4c7e57609f	physical:host	Oct. 26, 2020, 2:25 p.m.	Oct. 27, 2020, 6 p.m.	55.17
		CHI@UC	ba934192-dc3e-4e0a-917c-17d0bf629ee6	physical:host	Oct. 26, 2020, 10:05 a.m.	Oct. 28, 2020, 10:04 a.m.	95.97
		CHI@UC	c156bdd4-6f5b-40d9-ab27-86e7c6b64cb8	physical:host	Oct. 5, 2020, 10:40 a.m.	Oct. 6, 2020, 6 p.m.	31.33
		CHI@UC	ca9ba57f-4477-458a-af72-0ffc8fa187f3	physical:host	Oct. 5, 2020, 9:29 a.m.	Oct. 8, 2020, 6 p.m.	161.03
		CHI@UC	f2fb5aab-9681-422c-8ff0-61327c98c7da	physical:host	March 23, 2021, 2:44 p.m.	March 23, 2021, 4:48 p.m.	2.07
		CHI@UC	196fe530-f63a-41e2-bf31-4d6409587ae3	network	Oct. 26, 2020, 10:05 a.m.	Oct. 28, 2020, 10:04 a.m.	47.98

Fig. 6: Allocation usage details

To view the allocation usage details, in the *Allocations* section, click the *gear* button at the end of the allocation row, and then click *View Charge*. This will open a modal displaying a list of all charges against your allocation, including who initiated the charge, how many *Service Units* were charged, and what type of charge it was.

5.2.6 Manage Publications

To add publications to a project, click the *Add Publications* button in the *Project Details* page. Please enter the publications in BibTex format. All regular BibTex publication types are supported. If you can provide a link, please enter as *note* or *howpublished* using the url package.

To manage the publications you have entered, use the *Publications Dashboard*.

In the dashboard, you may remove a publication of a project by clicking the - button next to the publication text.

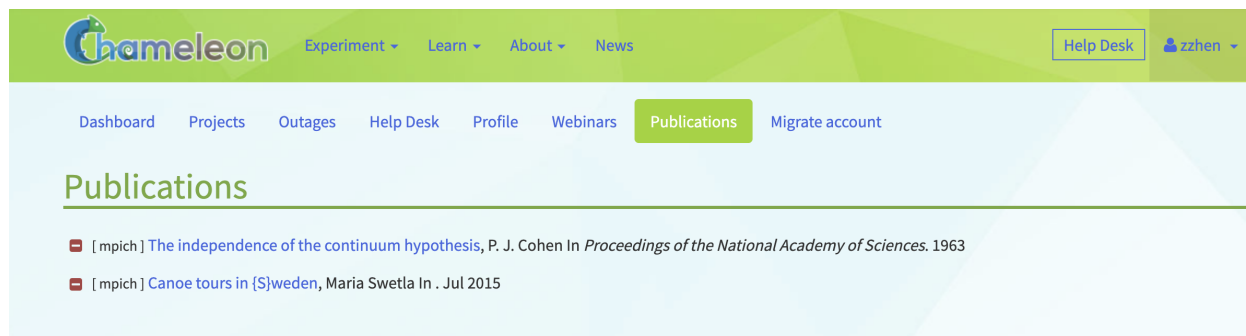


Fig. 7: Publications dashboard

5.2.7 Manage Users

In the dashboard, you can add or remove users (or “members”) from your projects, manage your project members’ user roles, and allocate how many SUs each project member can consume on your project allocation.

User Roles

To manage user roles for a *Project*, scroll down to the *Project Members* section in the *Project Details* page of your dashboard. The table below shows the types of roles that members can have and their privileges.

Role	Description
PI	Each project has only one PI. PI can manage roles of the project members.
Manager	Each project can have multiple Managers. Managers can manage project membership and renew allocations of the project.
Member	Members can only view the list of the project members.

To change the role of a project member, choose a new role from the dropdown and click the *Submit* button to apply the new role to the user, or use the *Cancel* button to cancel the action.

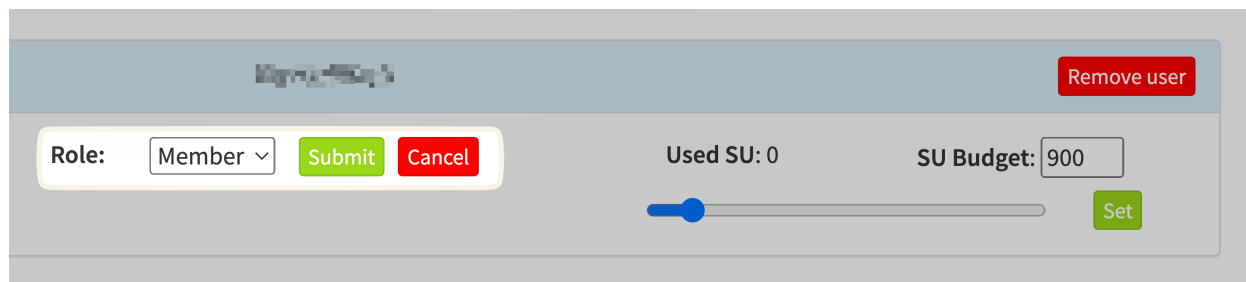


Fig. 8: Manage Role of a User

Set SU Budgets for Project Members

The PI or project managers can set a service unit (SU) budget for each project member. This budget limits the resources that a project member can utilize from the project's total SU allocation. Managing these budgets ensures fair distribution of resources and effective utilization of project resources across multiple project members. Setting a user-specific budget can help when managing resources for a project with lots of members (a large collaborative research project or a classroom project, for instance) and ensure that project allocations are shared effectively between project members.

Project managers (including PIs) can set a default SU budget that is applied to all project members except managers. All new users added to the project will receive the same default SU budget upon joining.

Viewing User SU Budgets - Project members will have their SU budget displayed next to their name in the *Project Members* table. This represents the allocation of resources that they can utilize within the project.

Project managers may also set different limits for different users. To adjust the SU budget for a specific user, use the slider or the SU Budget field to *Set* the new budget for an individual user.

Adding and Removing New Members

To add or remove users of a *Project*, use the *Project Members* section in the *Project Details* page on your dashboard.

You may add a user to your project by filling out their username or email address and clicking the *Add user* button. While each user has their own Chameleon User account independent of your project, they may be added to one or more projects. Being a user of a *Project* **does not** require a *PI eligibility*.

You may remove a user from your project by locating the user in the user list; clicking the *gear* button at the end of the row; and clicking *Remove user*.

It is also possible to bulk-add a large list of users by clicking the “Add multiple users” button, or remove all users without the Manager role by clicking the “Remove multiple users” button. Additionally, under this option there is a link which you can send to users that will allow them to request to join your project after they sign in to Chameleon. Once a request is made, the managers of a project will be notified, and will need to confirm the user.

If there is no user associated with an email address, an invitation will be sent with a link. When someone clicks on this link, they will be prompted to sign in or create an account, and then automatically added to the project. Invitations

Please enter usernames or emails below, one per line. For example

```
user1  
user2@uchicago.edu  
user3
```

Alternatively, you can send this link to users which will allow them to request to join your project.

<https://chameleoncloud.org/user/projects/join/request/9gJT7ImkcUfdnW9MEz3AIL-YZe/>

Username or emails*

Usernames one per line

Close

Add multiple users

Fig. 9: Adding multiple users

show up at the bottom of the members list, and can be deleted or resent if needed. After an invitation is accepted, the user will show up under the *Project Members* section.

USER PROFILE

The [Profile](#) page in the Dashboard allows you to manage your biographical information and membership to any Chameleon mailing lists.

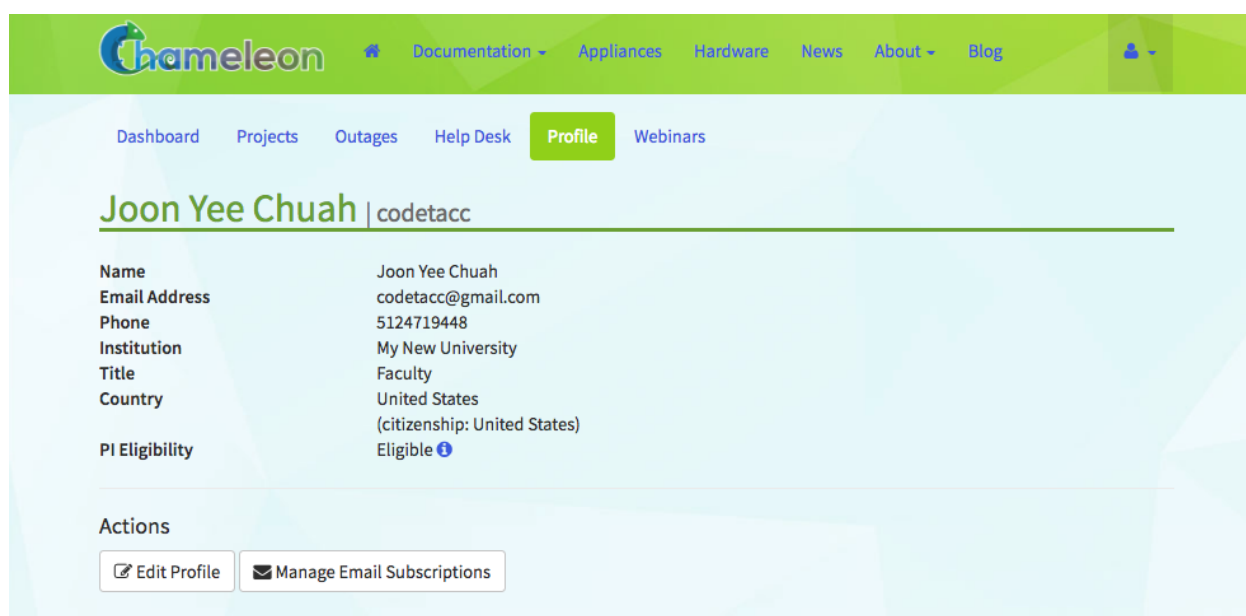


Fig. 1: The Profile page

To edit your biographical information, click the [Edit Profile](#) button.

GETTING HELP

7.1 Mailing lists

You can communicate with other Chameleon users by sending email to users@chameleoncloud.org. We also use this mailing list to communicate minor announcements or provide early access to some new hardware or features. All Chameleon users are registered by default: we recommend that you stay registered. However, if you really want to opt-out, you can do so via [your user profile](#).

7.2 Outages

The [Outages](#) page of the Dashboard contains a list of system outage announcements. You may subscribe to an RSS feed of these outages by clicking on the [RSS](#) icon.

Chameleon Documentation Appliances Hardware News About Blog

Dashboard Projects **Outages** Help Desk Profile Webinars

Reported Outages

[RSS](#)

CHI@UC maintenance for networking configuration changes

Resolved Posted by Pierre Riteau on January 31, 2018

Outage start	Tuesday, February 06, 2018 2 p.m.
Expected end	Wednesday, February 07, 2018 10 a.m.

Update: Issues at CHI@UC have been resolved. We apologize for any inconvenience.

--

A maintenance is scheduled on CHI@UC to apply networking configuration changes on Tuesday February 6, from 2 PM to 6 PM Central Time. All resources have been reserved. You will be able to use CHI@TACC in the meantime.

Fig. 1: The Outages announcement page

7.3 Help Desk

The [Help Desk](#) allows you to submit help request tickets and view the status of any open tickets.

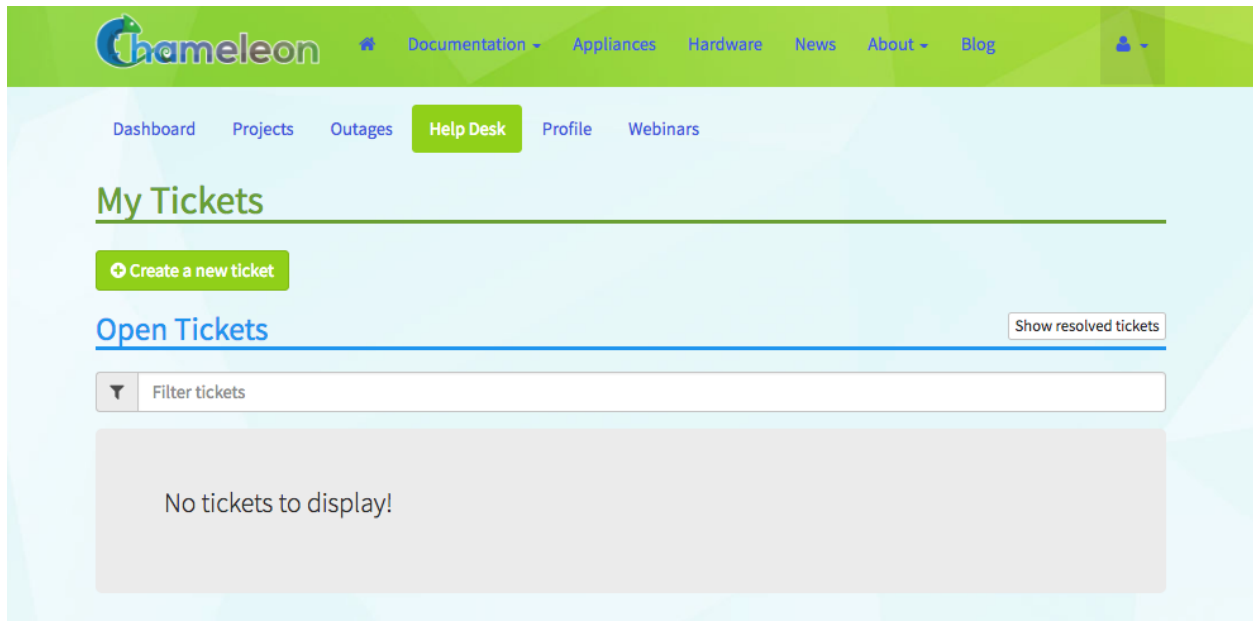


Fig. 2: The Help Desk page

To create a new help ticket, click the [+Create a new ticket](#) button and fill in the form. A system administrator will respond to your ticket within 3 business days.

Note: An alternative way of asking for help is sending an email to the [Chameleon users mailing list](#), especially when the Help Desk is down or you think it's something worth sharing with all Chameleon users. A system administrator will reply to your email and, if necessary, create a ticket for you.

7.4 Webinars

The [Webinars](#) page provides a list of upcoming webinars for Chameleon user training.

GRAPHICAL USER INTERFACE (GUI)

8.1 Introduction

The Graphical User Interface (GUI) provides a point-and-click experience for working with Chameleon resources. From the GUI, you may perform tasks such as manage and launch instances, and configure custom networking. Additionally, you may download an *OpenStack RC* file from the GUI if you wish to work with the *Command Line Interface*, instead. The Chameleon GUI is built on top of *OpenStack Horizon*. There are two Chameleon resource sites, each with its own URL (though it is possible to easily switch from one to other, see *Project and Site Menu*).

- The Texas Advanced Computing Center resources (CHI@TACC) are available at:

<https://chi.tacc.chameleoncloud.org>

- The University of Chicago resources (CHI@UC) are available at:

<https://chi.uc.chameleoncloud.org>

Chameleon also hosts an *OpenStack KVM* implementation where you may work with virtual machines. This site **does not** have access to bare metal resources. It is available at:

<https://kvm.tacc.chameleoncloud.org>

This section provides an overview of features available on the GUI for the bare metal sites at the Texas Advanced Computing Center (CHI@TACC) and the University of Chicago (CHI@UC). For information about *OpenStack KVM*, please see *KVM*.

You may login to either site using your Chameleon portal username and password.

Attention: Each Chameleon testbed sites—CHI@TACC, CHI@UC, and KVM@TACC—are **independent**, so snapshots, keypairs, Swift containers, Gnocchi metrics and other objects are unique to each site. For example, a keypair created at the CHI@TACC site is **not** available at the CHI@UC site. In addition, the bare metal resource types vary between sites.

8.2 GUI Features

Upon logging in to the GUI at a Chameleon site, you will see your project's Overview page.

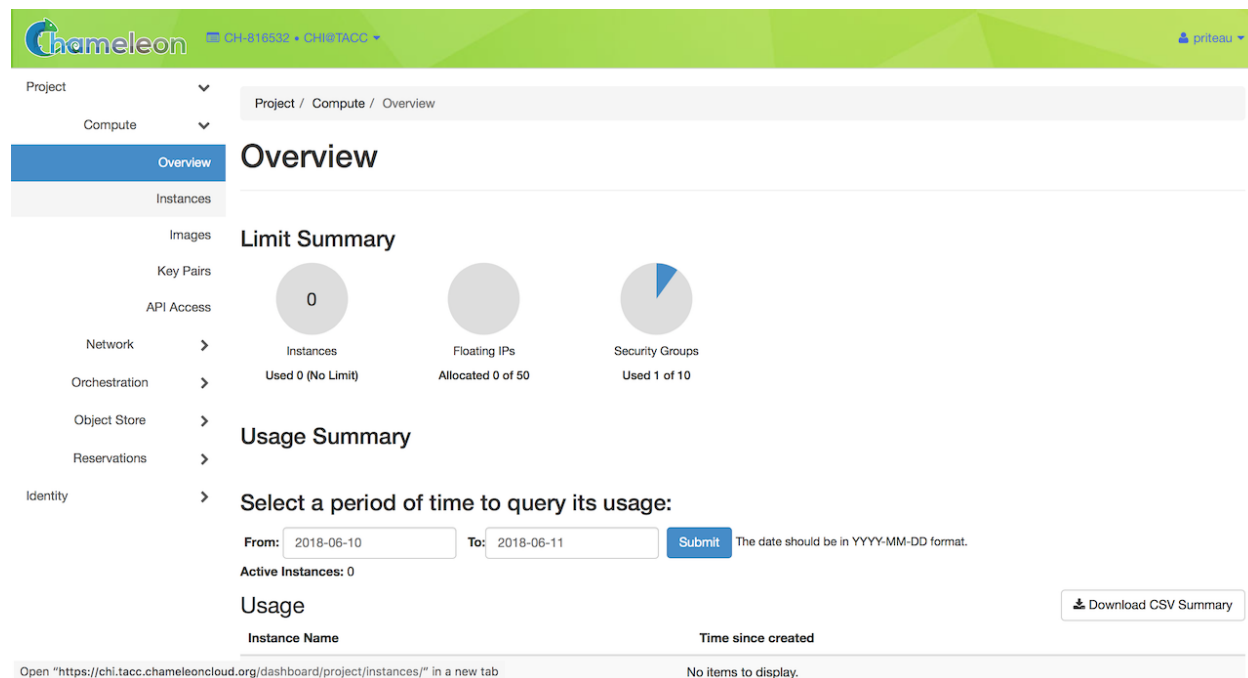


Fig. 1: The Chameleon GUI

8.2.1 Project and Site Menu

To switch among the projects you belong to, use the project and site menu—the dropdown on the upper left of the screen next to the Chameleon logo. You can also use this menu to switch from one Chameleon site to another. This allows you to easily perform multi-site experiments.

8.2.2 User Menu

To access user specific settings and download *OpenStack RC* files, use the user menu—the dropdown on the upper right of the screen where you will see your account name.

8.2.3 Settings

In the settings menu, you can change user specific settings such as the Timezone.

Note: Updating your timezone is **highly** recommended. When you make reservations for bare metal resources, your local time will be used. UTC is the default Timezone.

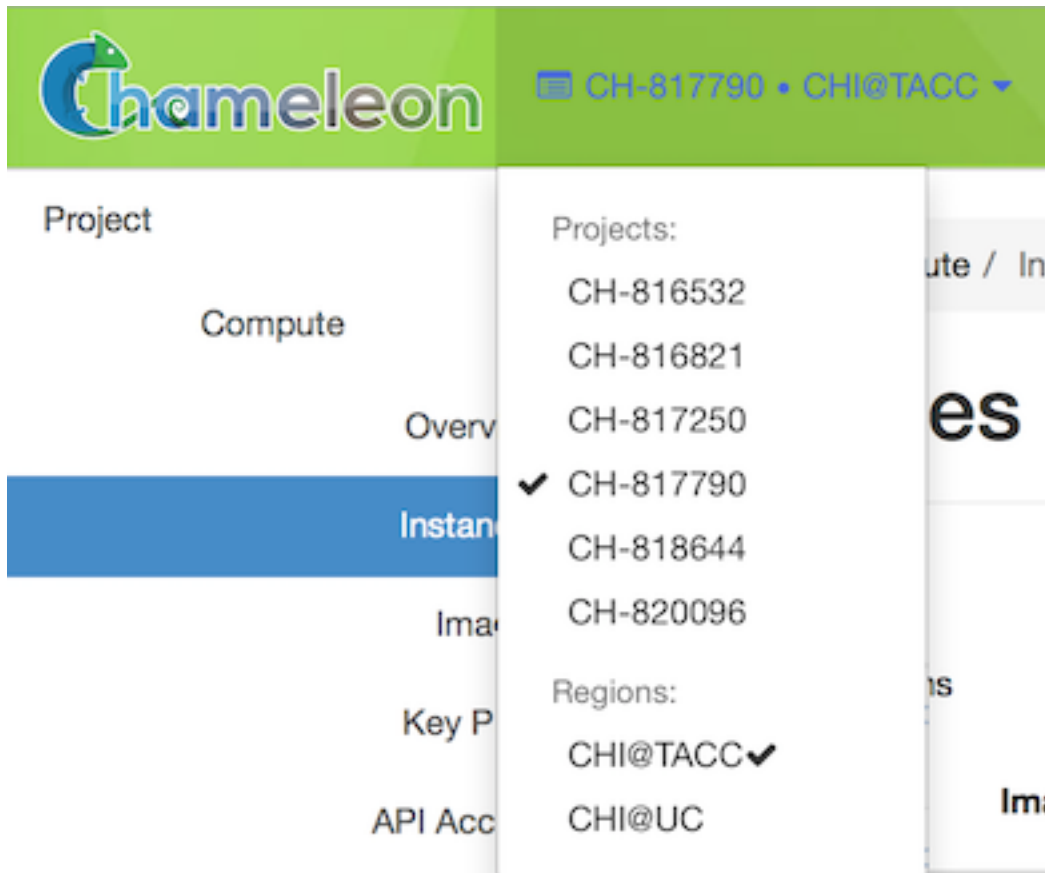


Fig. 2: Switching between projects

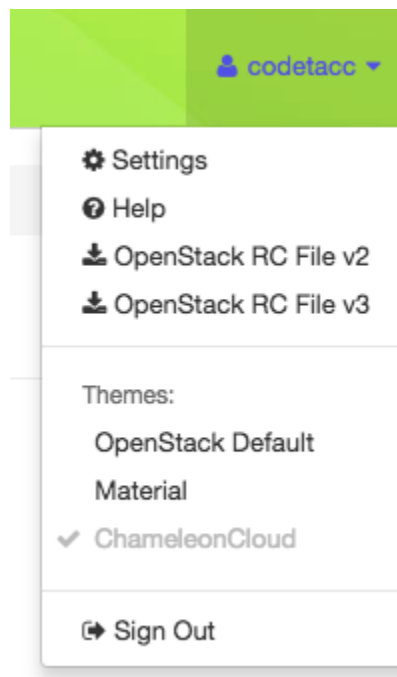


Fig. 3: The user dropdown menu

Project > Settings / User Settings

Settings > User Settings

User Settings

Change Password

Identity >

User Settings

Language *
English (en) ⌵

Timezone *
UTC ⌵

Items Per Page * ⓘ
20

Log Lines Per Instance * ⓘ
35

Description:
Modify dashboard settings for your user.

Save

Fig. 4: User settings

8.2.4 Help

The *Help* menu item will take you to this documentation site.

8.2.5 OpenStack RC File

Clicking on this menu items will download a customized [RC file](#) for use with the OpenStack Command Line Interface. Source the RC file using `source` command to configure environment variables that allow you to easily log in using the *Command Line Interface*. For more information about *OpenStack RC* script, please see [The OpenStack RC Script](#).

8.2.6 Themes

You may change the GUI theme by selecting the provided menu items.

8.2.7 Sign Out

Use the *sign out* menu item to sign out from your current site.

Note: If you do not sign out manually, your session will expire in one hour.

8.3 Navigating the GUI

The navigation sidebar allows you to access different sections.

8.4 API Access

The API Access page lists all the available REST APIs that are used for configuring the *Command Line Interface (CLI)*. In addition, you may download *The OpenStack RC Script* scripts via this page.

Note: Typically, the key generated from your computer will be at `~/.ssh/id_rsa.pub`. On Mac OS X, you can run in a terminal: `cat ~/.ssh/id_rsa.pub | pbcopy`. It copies the content of the public key to your copy/paste buffer. Then you can simply paste in the “Public Key” box.

8.5 Compute

Use *Compute* section for reserving, configuring and managing your instances.

8.5.1 Overview

The Overview page provides a graphical summary of your project’s current resource usage.

Note: At the bare metal sites, you may launch as many instances as you like, but bounded by the project *Service Unit* allocation. However, at the OpenStack KVM site, your project is limited to a certain number of virtual machines. By default, each project is allowed to allocate 50 *Floating IP addresses* and use 10 *Security Groups*. You may request additional resources by submitting a ticket on the [Help Desk](#).

8.5.2 Instances

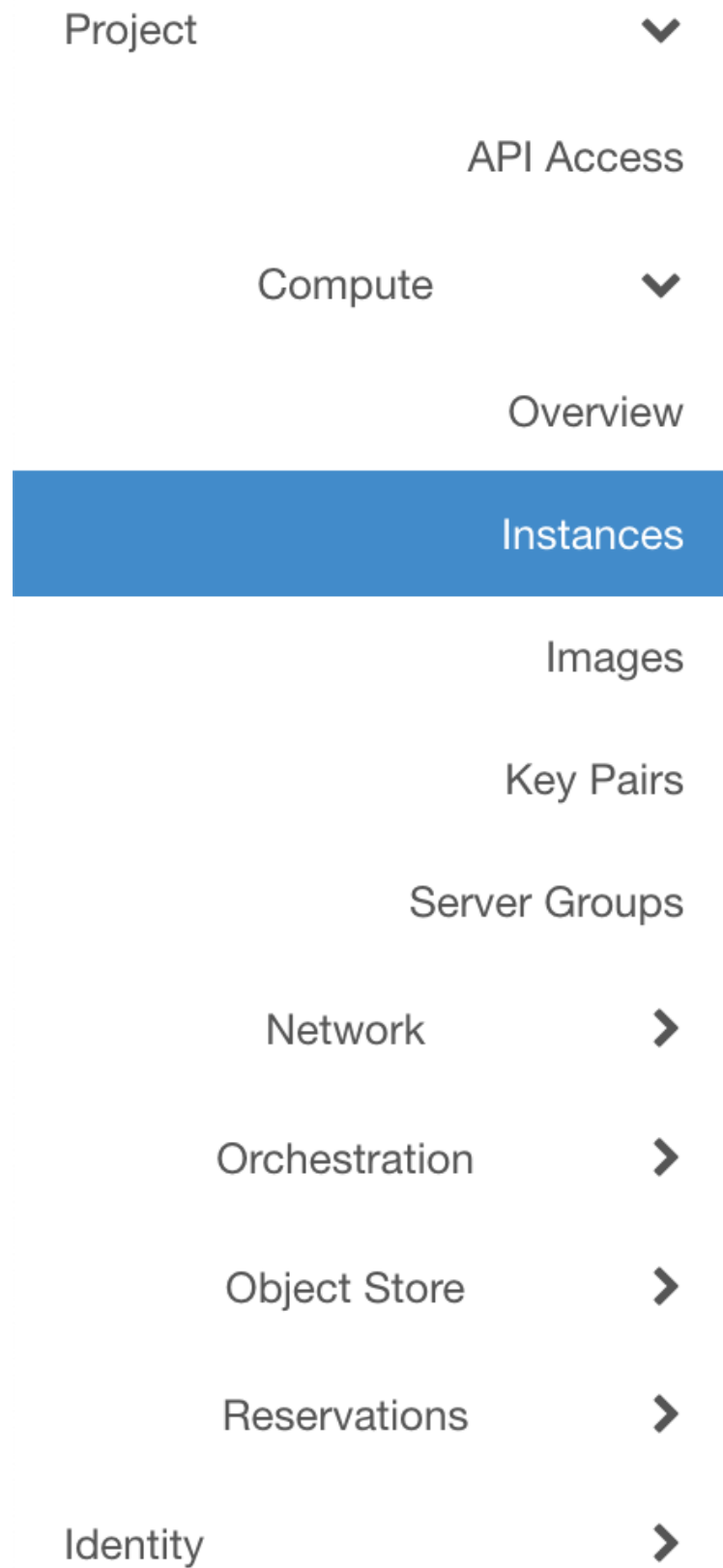
The Instances page allows you to work with your instances. You may launch, terminate, monitor, or reboot an instance. Clicking on the dropdown list in *Action* column to see what you are eligible to do to your instances.

8.5.3 Images

The Images page allows you to view, upload and edit the images. You may also use this page to launch instance using selected images.

Note: You can only edit the images you own.

Tip: Search for images using the filter bar.



Project / API Access

API Access

View Credentials

Download OpenStack RC File ▾

Displaying 17 items

Service	Service Endpoint
Baremetal	https://chi.uc.chameleoncloud.org:6385
Baremetal Introspection	-
Cep	https://chi.uc.chameleoncloud.org:8910
Cloudformation	https://chi.uc.chameleoncloud.org:8000/v1
Compute	https://chi.uc.chameleoncloud.org:8774/v2.1
Compute_Legacy	https://chi.uc.chameleoncloud.org:8774/v2/975c0a94b784483a885f4503f70af655

Fig. 5: The API Access page


Project ▾
Compute ▾
Overview
Instances
Images
Key Pairs
API Access
Network >
Orchestration >
Object Store >
Reservations >
Identity >


Project / Compute / Overview

Overview

Limit Summary

0
Instances
Used 0 (No Limit)


Floating IPs
Allocated 1 of 50


Security Groups
Used 2 of 10

Usage Summary

Select a period of time to query its usage:

From: 2018-02-20

To: 2018-02-21

Submit

The date should be in YYYY-MM-DD format.

Active Instances: 0

Project

Compute

Overview

Instances

Images

Key Pairs

API Access

Network

Orchestration

Project / Compute / Instances

Instances

Instance ID = Filter

Instance Name	Image Name	IP Address	Flavor	Key Pair	Status	Availability Zone	Task	Power State	Time since created	Actions
No items to display.										

Project

Compute

Overview

Instances

Images

Key Pairs

API Access

Network

Orchestration

Object Store

Reservations

Identity

Project / Compute / Images

Images

Click here for filters.

Displaying 20 items

123456

<input type="checkbox"/>	Name ^	Type	Status	Visibility	Protected	Disk Format	Size	
<input type="checkbox"/>	> ARM-IPA-kernel	Image	Active	Public	No	AKI	9.41 MB	
<input type="checkbox"/>	> ARM-IPA-ramdisk	Image	Active	Public	No	ARI	270.55 MB	
<input type="checkbox"/>	> ARM64-CC-Ubuntu16.04	Image	Active	Public	No	QCOW2	633.19 MB	<div>Launch</div>

8.5.4 Key Pairs

The Key Pairs page allows you to create, import and manage SSH key pairs associated with your user account.

Project / Compute / Key Pairs

Key Pairs

Displaying 2 items

<input type="checkbox"/>	Key Pair Name	Fingerprint	Actions
<input type="checkbox"/>	cctrain	3c:cd:d2:09:51:a1:17:50:31:ec:ba:e4:6d:2c:80:d3	Delete Key Pair
<input type="checkbox"/>	snapshotdemo	7d:70:5a:60:04:f6:eb:72:db:d4:0b:30:0c:83:6f:29	Delete Key Pair

Displaying 2 items

Note: Chameleon **only** stores the *public key* for each SSH key pair. **Do not** upload your *private key* to the portal! Private keys look like this:

```
-----BEGIN RSA PRIVATE KEY-----
```

To delete a SSH key pair, click on the *Delete Key Pair* button in the *Action* column. You may delete multiple key pairs by selecting them via the checkbox and clicking the *Delete Key Pairs* button.

Creating a Key Pair

To create a key pair, click the + *Create Key Pair* button. In the prompted dialog, provide a name for your new key pair and then click the *Create Key Pair* button.

Create Key Pair

Key Pair Name *

Key pairs are SSH credentials which are injected into images when they are launched. Creating a new key pair registers the public key and downloads the private key (a .pem file).

Protect and use the key as you would any normal SSH private key.

[Cancel](#) [Create Key Pair](#)

Fig. 6: Specifying a key pair name

A .pem file that contains the *Private Key* should be automatically downloaded. In addition, the *Public Key* associated with the *Private Key* should be saved automatically to Chameleon. Clicking on the *Regenerate and download Key Pair*

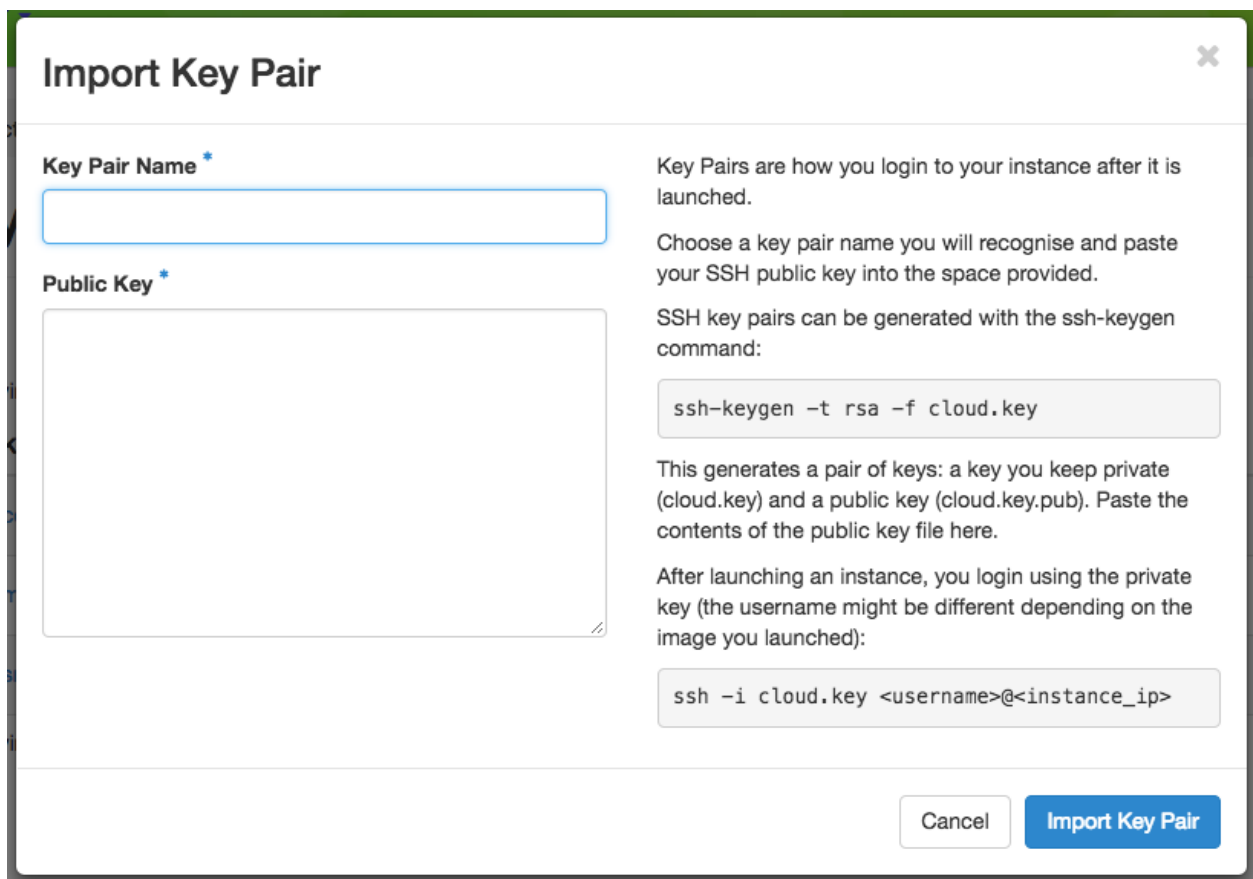
button will generate a new *Public/Private Key Pair* and initiate a new download of the *Private Key*.

Tip: Save the *Private Key* to a location you will remember at your local file system. Your *home* directory is recommended for macOS and Linux systems.

Importing a Key Pair

Alternatively, you may import a key pair that you have generated on your computer. Clicking the *Import Key Pair* button to prompt the dialog. Then, provide a name for your imported key pair and paste the *Public Key*.

Tip: The prompted dialog contains the instructions on how to generate a key pair using the Linux/macOS command.



Import Key Pair

Key Pair Name *

Public Key *

Key Pairs are how you login to your instance after it is launched.

Choose a key pair name you will recognise and paste your SSH public key into the space provided.

SSH key pairs can be generated with the ssh-keygen command:

```
ssh-keygen -t rsa -f cloud.key
```

This generates a pair of keys: a key you keep private (cloud.key) and a public key (cloud.key.pub). Paste the contents of the public key file here.

After launching an instance, you login using the private key (the username might be different depending on the image you launched):

```
ssh -i cloud.key <username>@<instance_ip>
```

Cancel Import Key Pair

Fig. 7: Importing a public key

Tip: Typically, the key generated from your computer will be at `~/.ssh/id_rsa.pub`. On Mac OS X, you can run in a terminal: `cat ~/.ssh/id_rsa.pub | pbcopy`. It copies the content of the public key to your copy/paste buffer. Then you can simply paste in the “Public Key” box.

8.6 Network

The Network section allows you to work with virtual network resources, such as configuring routers and virtual networks. For more information, please see [Networking](#).

8.6.1 Network Topology

The Network Topology page displays your current virtual network topology in either the *Topology* or *Graph* formats. You may also use this section to directly launch instances, create networks or create routers.

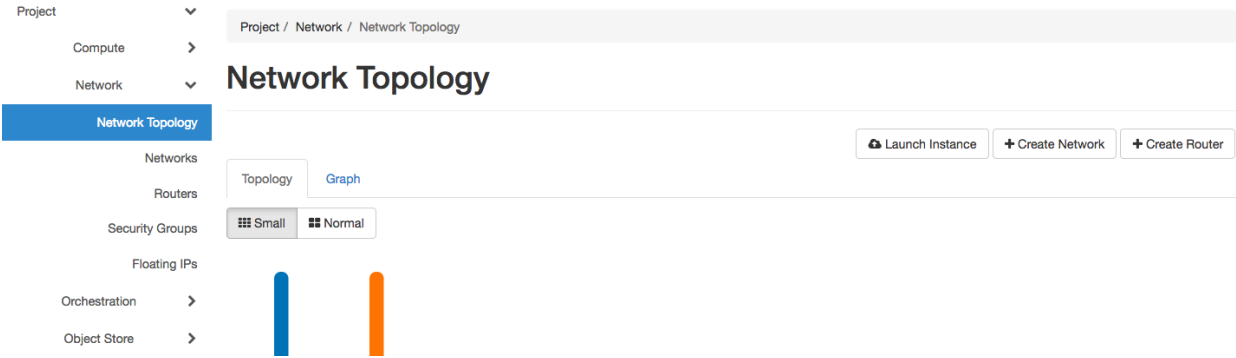


Fig. 8: The Network Topology page

8.6.2 Networks

The Networks page lists all the Virtual Networks of the selected project. You may use this section to create, delete and modify Virtual Networks. Clicking on the dropdown list (if shown) in *Action* column to see what you are eligible to do to your virtual networks.

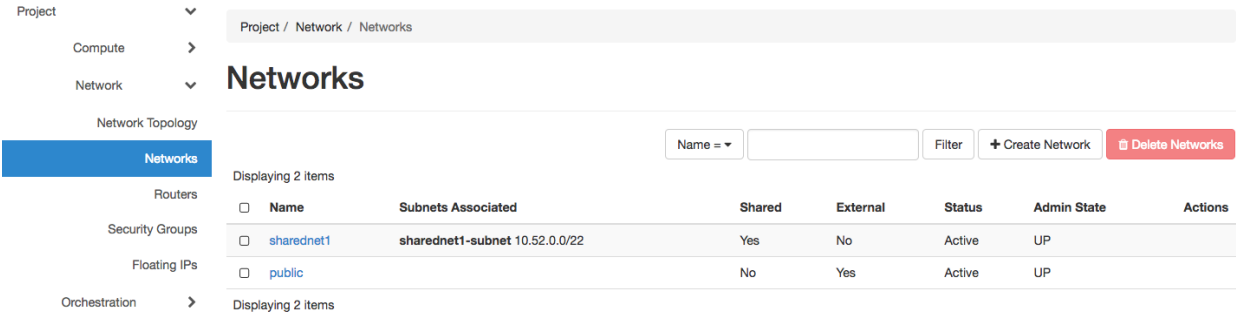


Fig. 9: The Networks page

8.6.3 Routers

Same as the Networks page, the Routers page allows you to work on the Routers of the selected project.

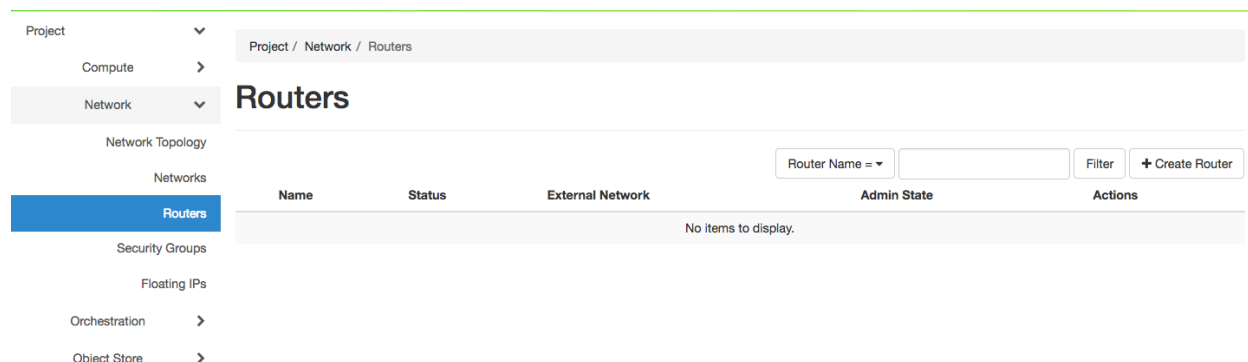


Fig. 10: The Routers page

8.6.4 Security Groups

Use the Security Groups page to create, delete, and modify the Security Groups of the selected project.

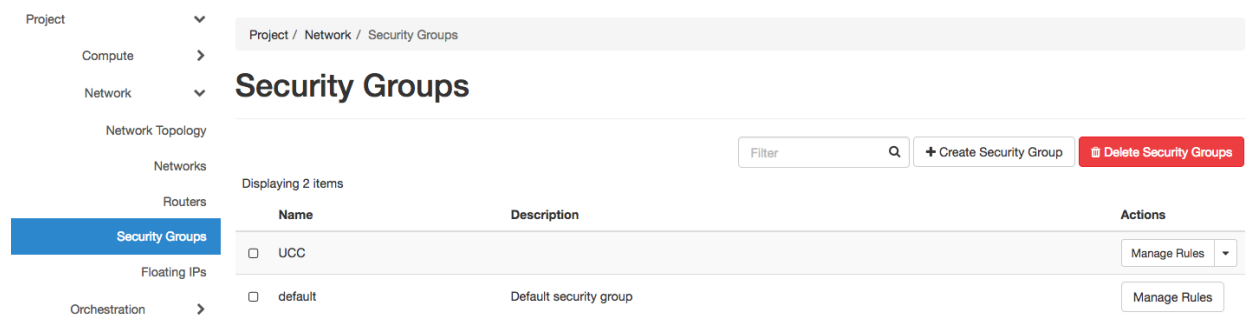


Fig. 11: The Security Groups page

Attention: Chameleon bare metal sites—[CHI@TACC](#) and [CHI@UC](#)—do not support security groups, i.e., all ports are open to the public.

8.6.5 Floating IPs

The Floating IPs page allows you to work with the Floating IP addresses allocated for the selected project, including associating with instances and releasing back to the pool. Clicking on the dropdown list (if shown) in *Action* column to see what you are eligible to do to your Floating IPs.

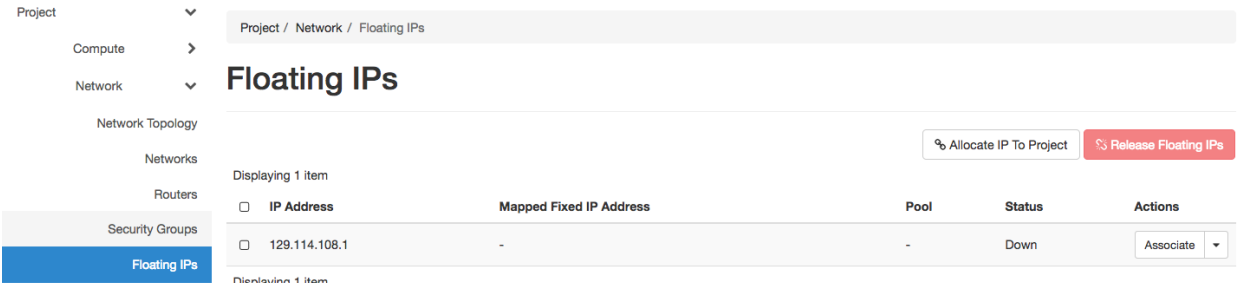


Fig. 12: The Floating IPs page

Releasing Floating IP Addresses

Important: The Chameleon Floating IP address pool is a shared and finite resource. **Please be responsible and release the Floating IP addresses that are not used, so other Chameleon users and projects can use them!**

To release a single Floating IP address, click on the dropdown in the *Actions* column and select *Release Floating IP* . You may also release multiple addresses by selecting them via checkboxes and clicking the *Release Floating IPs* button.



Fig. 13: Releasing a Floating IP address

8.7 Orchestration

The Orchestration section allows you to work with the *Chameleon's Complex Appliances*.

8.7.1 Stacks

A deployed complex appliance is referred to as a “stack” – just as a deployed single appliance is typically referred to as an “instance”. The Stacks page allows you to launch, rebuild, or terminate stacks.

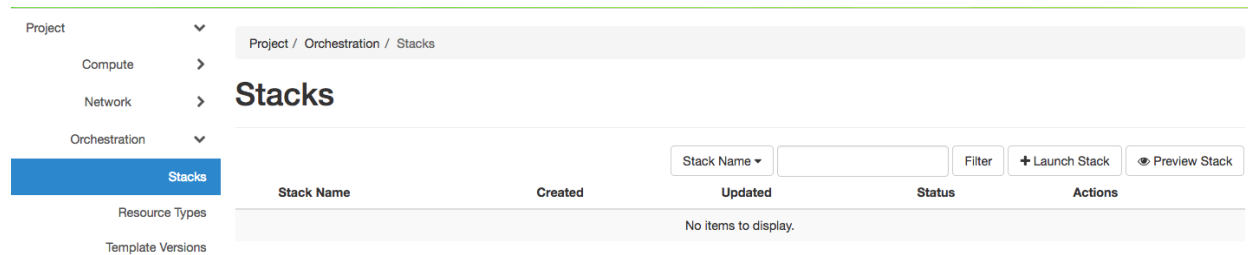


Fig. 14: The Stacks page

Tip: After launching a stack, all the instances launched with the stack can be viewed at *Compute - Instances* section as well.

Note: When you terminate a stack, all instances launched with the stack will be terminated.

8.7.2 Resource Types

The Resource Types page lists the currently available Orchestration Resource Types of Chameleon. You may click on the resource types to get details. The Orchestration Resource Types are used when writing *OpenStack Heat Orchestration Template*. For more information about *OpenStack Heat*, please see the [OpenStack Heat documentation](#).

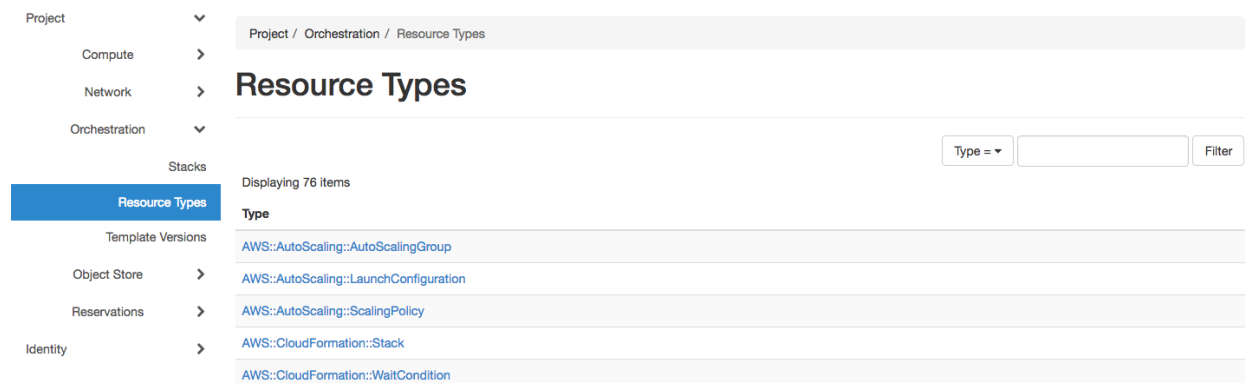


Fig. 15: The Resource Types page

8.7.3 Template Versions

The Template Versions are also used when writing *OpenStack Heat Orchestration Template*. Clicking on the version to get supported features of the specific version.

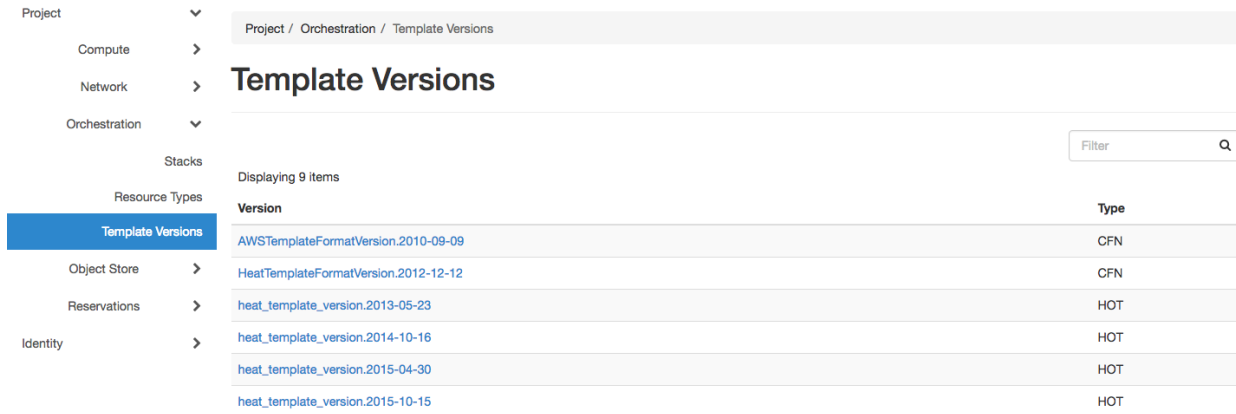


Fig. 16: The Template Versions page

8.8 Object Store

The *Containers* section under *Object Store* gives an easy access to your Chameleon object/blob store. You may create, delete, upload objects to or remove objects from containers via this page. For more information about Chameleon Object Store, please see [Object Store](#).

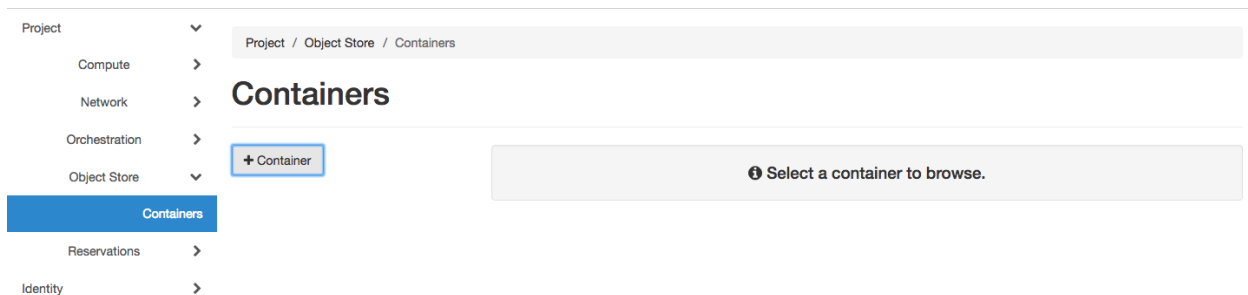


Fig. 17: The Containers page

8.9 Reservations

The Reservations section allows you to manage your leases of the selected project, including creating and deleting leases. For more information, see [Reservations](#).

Tip: Check *Lease Calendar*, so you can schedule your experiments efficiently.

Project

Compute

Network

Orchestration

Object Store

Reservations

Leases

Identity

Project / Reservations /

Leases

Lease Calendar

Create Lease

Delete Leases

Displaying 8 items

<input type="checkbox"/>	Lease name	Start date	End date	Action	Status	Reason	Actions
<input type="checkbox"/>	ucc_gpu_power_m40	2017-12-02 14:00 UTC	2017-12-06 23:00 UTC	STOP	COMPLETE	Successfully stopped lease	Delete Lease
<input type="checkbox"/>	ucc_gpu_power	2017-12-02 14:00 UTC	2017-12-06 23:00 UTC	STOP	COMPLETE	Successfully stopped lease	Delete Lease
<input type="checkbox"/>	chameleon_lease_jchuah	2017-10-02 16:05 UTC	2017-10-03 16:04 UTC	STOP	COMPLETE	Successfully stopped lease	Delete Lease
<input type="checkbox"/>	ucc_compute_node	2017-12-05 19:31 UTC	2017-12-06 19:30 UTC	STOP	COMPLETE	Successfully stopped lease	Delete Lease

Fig. 18: The Leases page

8.10 Identity

The Project section under Identity allows you to check what projects you belong to. You can set your default project by clicking the *Set as Active Project* button in the *Actions* column.

Project

Identity

Projects

Identity / Projects

Projects

Project Name =

Filter

Displaying 1 item

<input type="checkbox"/>	Name	Description	Project ID	Domain Name	Enabled	Actions
<input type="checkbox"/>	CH-819507		d5233415ee0b467baec14cbd2d0e1331	Default	Yes	

Displaying 1 item

Fig. 19: The Projects page

COMMAND LINE INTERFACE (CLI)

9.1 Introduction

The Command Line Interface (CLI) provides a way to interact with Chameleon resources using shell and scripting tools. Chameleon uses the [OpenStack Client](#) to provide CLI functionality. This documentation section provides an overview on how to install the client and configure your shell environment to access Chameleon features.

Attention: Some of the Chameleon features are **only** accessible via the CLI, such as the Gnocchi metrics and the advanced networking features.

Note: Chameleon Cloud is primarily designed to support Unix-like environments. Therefore, it is highly recommended using CLI in a Unix-like system. For Windows 10 users, you may want to enable [Windows Subsystem for Linux](#) to get better experience with the Chameleon CLI.

9.2 Installing the CLI

9.2.1 Prerequisites

1. **Python** - Check if you have Python installed.
2. **pip** - If you're using Python 3.4 (or greater), then pip comes installed with Python by default.

9.2.2 OpenStack Client Installation

1. Install the CLI by typing `pip install python-openstackclient` in the terminal.
2. Verify that it has installed correctly by typing `openstack`. You will enter the Openstack Client in interactive mode and your prompt should change to `(openstack)`.
3. Exit the client by typing `exit`.
4. There are some clients with new features or bugfixes not yet in the upstream release branches, notably the Blazar CLI client. If you want to make reservations via the CLI, you should install that here:

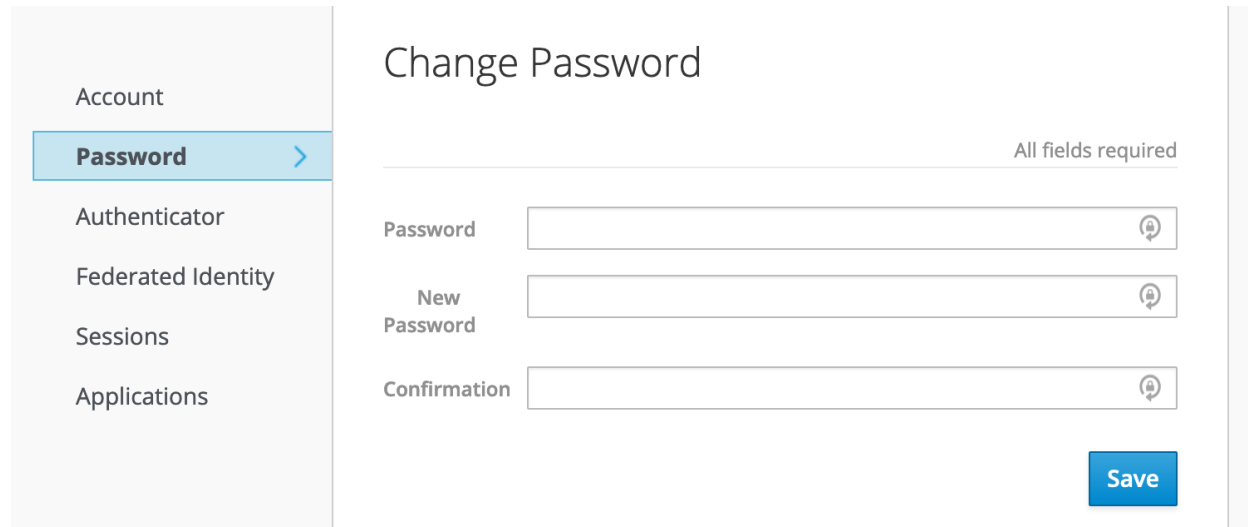
```
pip install git+https://github.com/chameleoncloud/python-  
↪blazarclient@chameleoncloud/xena
```

9.3 CLI authentication

When using the CLI, you have to provide some credentials so the system trusts that the operations are really being executed by your user account. There are two ways of doing this.

9.3.1 Setting a CLI password

You can set a CLI password via the [Chameleon Authentication Portal](#). The password you associate with your account can not be used to log in to the GUI or Jupyter interfaces and can only be used to authenticate a command-line client.



The screenshot shows the 'Change Password' interface in the Chameleon Authentication Portal. On the left is a sidebar menu with options: Account, Password (selected and highlighted in blue), Authenticator, Federated Identity, Sessions, and Applications. The main content area is titled 'Change Password' and includes a note 'All fields required'. There are three input fields: 'Password', 'New Password', and 'Confirmation', each with a password icon on the right. A blue 'Save' button is located at the bottom right of the form.

Fig. 1: Setting a password in the Chameleon Authentication Portal

The benefit of this method is that this password will work on any Chameleon site.

Note: You should set a strong password for your CLI password, and it should not be a password you use elsewhere. Otherwise, your account risks being compromised by an attacker who has possibly obtained your password from another breached service. We **highly** recommend using a password manager e.g., [BitWarden](#), [LastPass](#) <<https://www.lastpass.com/password-manager>>, or [1Password](#) to assist.

9.3.2 Creating an application credential

You can also generate *application credentials*, which act as dedicated one-off passwords that are authorized with the same permissions as your user account, within a single project. If you work on multiple projects simultaneously, you will need to generate one application credential for each project.

To create an application credential, navigate to the “Identity” dashboard in the [Graphical User Interface \(GUI\)](#), and go to the “Application Credentials” panel. Create a new application credential and name it something meaningful (such as “CLI access for project CH-XXX”). **You will also need to check the “unrestricted” checkbox in order to use the CLI to make leases in Blazar.** If you do not need to make reservations via the CLI, you can leave the box unchecked, as it is the safer option.

Once the system generates the credential, you will be given the option to download an [RC file](#) that configures the CLI to use the application credential for authentication. You will only see the secret credentials once, so make sure to save the RC file or the secret somewhere, as if it’s lost, you will have to delete the credential and create a new one.

Create Application Credential ✕

Name *

CLI access for CH-123456

Description

This is used to reserve resources, so the "unrestricted" checkbox is checked.

Secret

Expiration Date

mm / dd / yyyy

Expiration Time

-- : -- --

Roles

member

☒ **Unrestricted (dangerous)**

Description:

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".

Cancel

Create Application Credential

9.4 The OpenStack RC Script

You must use the *OpenStack RC Scripts* to configure the environment variables for accessing Chameleon features. You can download the script from the Chameleon GUI at the [API Access](#).

Hint: If you use the Chameleon supported (CC) images, you'll find an `openrc` file with a service token in the home directory for the `cc` user. The file will be auto-sourced when you login, so you can use the *openstack* and the *swift* CLI directly, as well as the *cc-snapshot* and the *cc-cloudfuse* tools.

1. Log in to the GUI at [CHI@TACC](#) or [CHI@UC](#).

Important: Download the RC file from the site you would like to interact with. The RC files are different for each site.

2. Select the project you wish to access via *Project and Site Menu*.

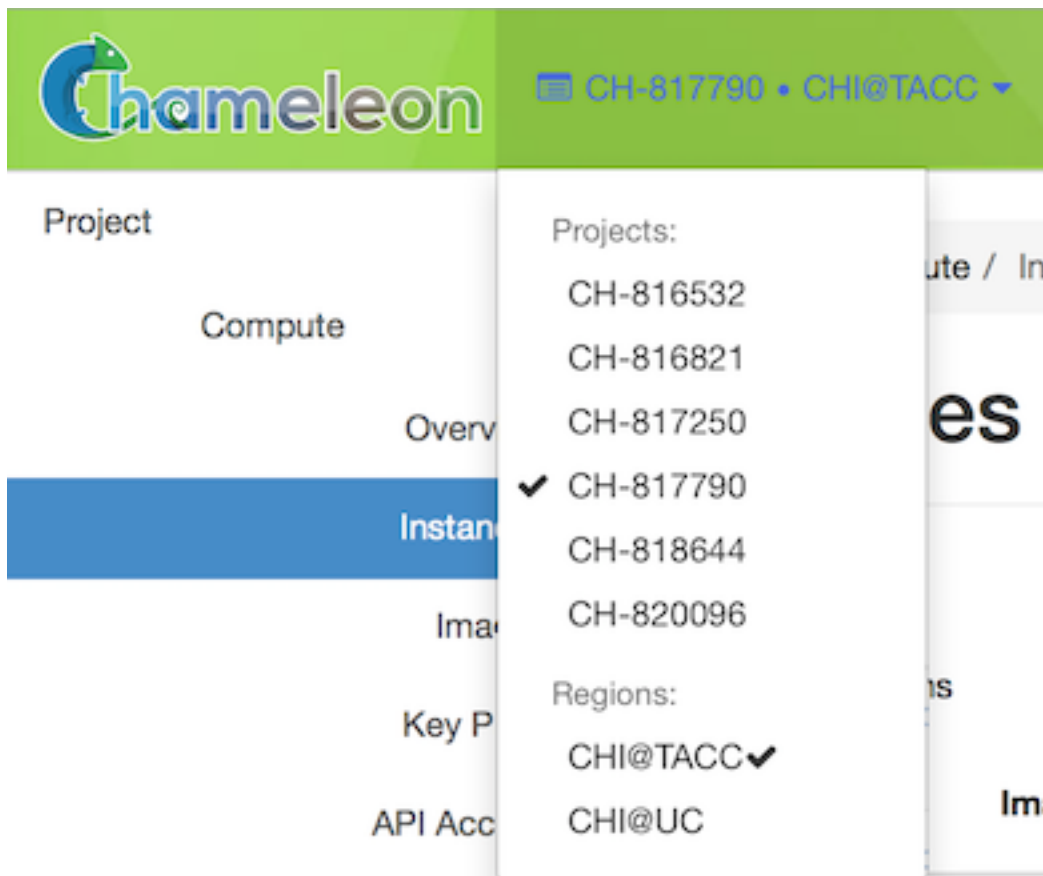


Fig. 2: The Project Dropdown

3. Download *OpenStack RC Script* using *User Menu* by clicking on *Openstack RC File v3*.
4. Run the following command in the terminal:

```
source <path/to/openstack_rc_file>
```

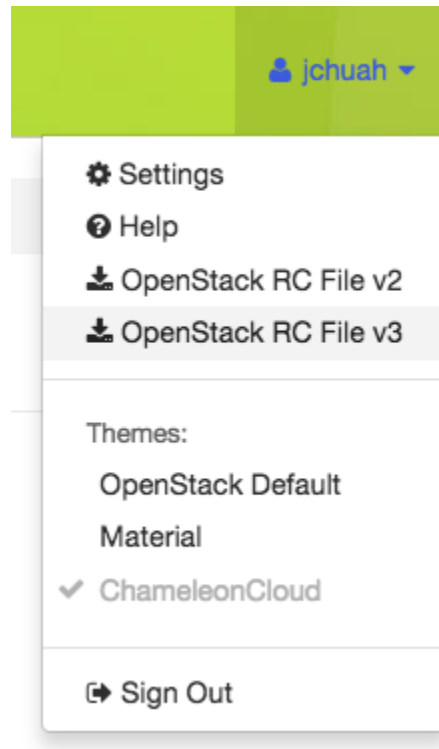


Fig. 3: The OpenStack RC File v3 link in the User Dropdown

Note: The command **will not** work for Windows users. Skip this step and the next step if you are using Windows system.

5. Enter your password when prompted.
6. For macOS/Linux users, your current terminal session has been configured to access your project. Now type `openstack` in your terminal session.

For Windows users, you have to provide the environment variables in the *OpenStack RC* script as `openstack` command parameters. Run the following command in your Windows prompt:

```
openstack --os-auth-url <OS_AUTH_URL> \
--os-project-id <OS_PROJECT_ID> \
--os-project-name <OS_PROJECT_NAME> \
--os-user-domain-name <OS_USER_DOMAIN_NAME> \
--os-username <OS_USERNAME> \
--os-password <OS_PASSWORD> \
--os-region-name <OS_REGION_NAME> \
--os-interface <OS_INTERFACE> \
--os-identity-api-version <OS_IDENTITY_API_VERSION>
```

Replace values of the parameters by reading from the *OpenStack RC* script.

Another way to configure the OpenStack client for Windows users is to add/edit environment variables manually via *System Properties* window. Then, click on *Environment Variables...* button and manually add/edit the environment variables in *OpenStack RC Script* to *Environment Variable* window.

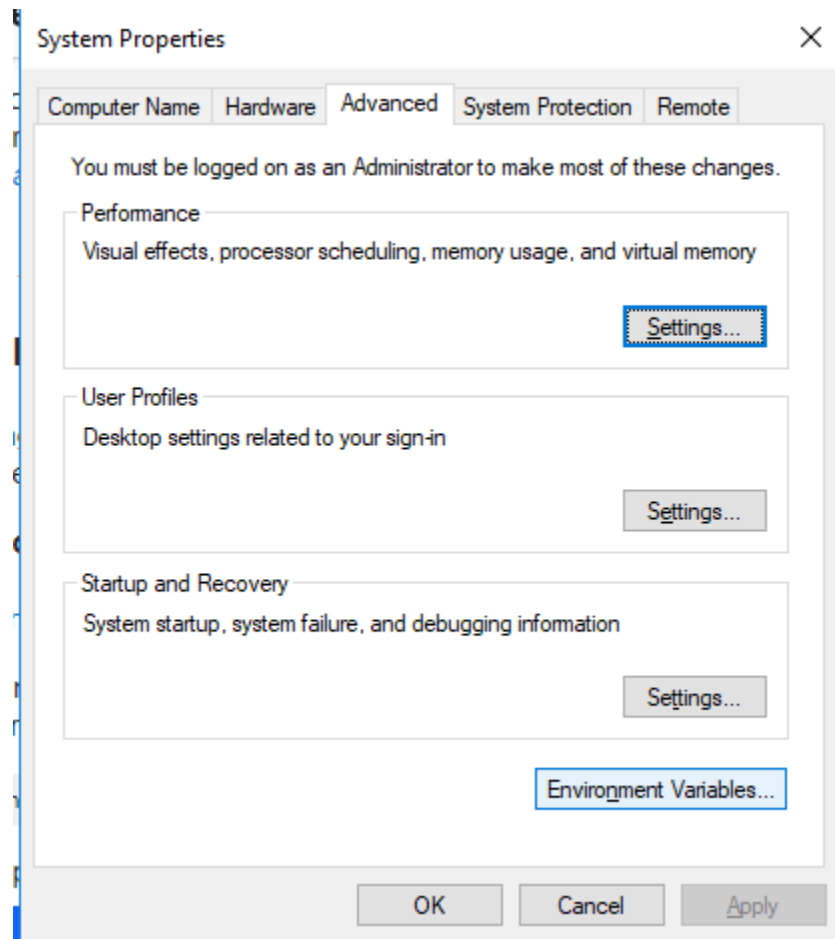


Fig. 4: System Properties Window of Windows System

Note: For macOS/Linux users, every time when open a new terminal, you have to run the `source` command to access the OpenStack client.

Error: If you get authentication error, check if you input your password correctly.

7. Type `project list` at the (openstack) prompt. You should see a list of the projects you belong to.

Error: If you get permission error at this step, please check that:

- the terminal session has been configured correctly with the environment variables
- the *OpenStack RC* script you `source` is **v3**
- the OpenStack client version is the latest. To check the OpenStack client version, use `openstack --version` command. Some older versions may cause errors.

Error: If you get the `Missing value` error when using a command, it is likely that your terminal session has not been configured correctly and completely with the environment variables. The error may be fixed by re-running the `source` command over the OpenStack RC Script or using the command line switches.

9.5 Using the CLI

You can use the CLI in either Interactive Mode or Shell Mode. In either mode, the OpenStack client has to be configured by using the *OpenStack RC Script* or by providing the command line switches. For more information about the usage of the OpenStack client, run `openstack --help`.

9.5.1 Interactive Mode

The Interactive Mode allows you to use the `openstack` commands through an interactive prompt. To start the Interactive Mode, type `openstack` in the configured terminal. Once entering the Interactive Mode, you will see a (openstack) prompt. Type the command you would like to run at the prompt. To find out the commands, type `help`.

9.5.2 Shell Mode

Each CLI command can be used in your terminal exactly the same way that it appears in the Interactive Mode, simply by preceding the command with `openstack`. For example, the command `image list` in the Interactive Mode is equivalent to the command `openstack image list` in the Shell Mode.

JUPYTER INTERFACE

Jupyter Notebooks are an excellent tool for prototyping, exploring, and ultimately documenting the entire experimental process. They combine the benefits of explanatory text, executable code, and rich visualization/interaction.

Chameleon users can get a Jupyter Notebook server automatically provisioned for them by logging in to the [JupyterHub server](#) managed by Chameleon. Upon login, you will be redirected to your Jupyter Notebook server. If there is not yet a Notebook server allocated for your user, one will be created behind the scenes. This can take a few moments.

Warning: The shared Jupyter environment places resource limits on your Jupyter server, notably limiting it to 1 CPU core and 1GB of memory. If you are doing computationally or memory-intensive work in a Notebook, it may be beneficial to look in to [provisioning your own dedicated JupyterHub](#).

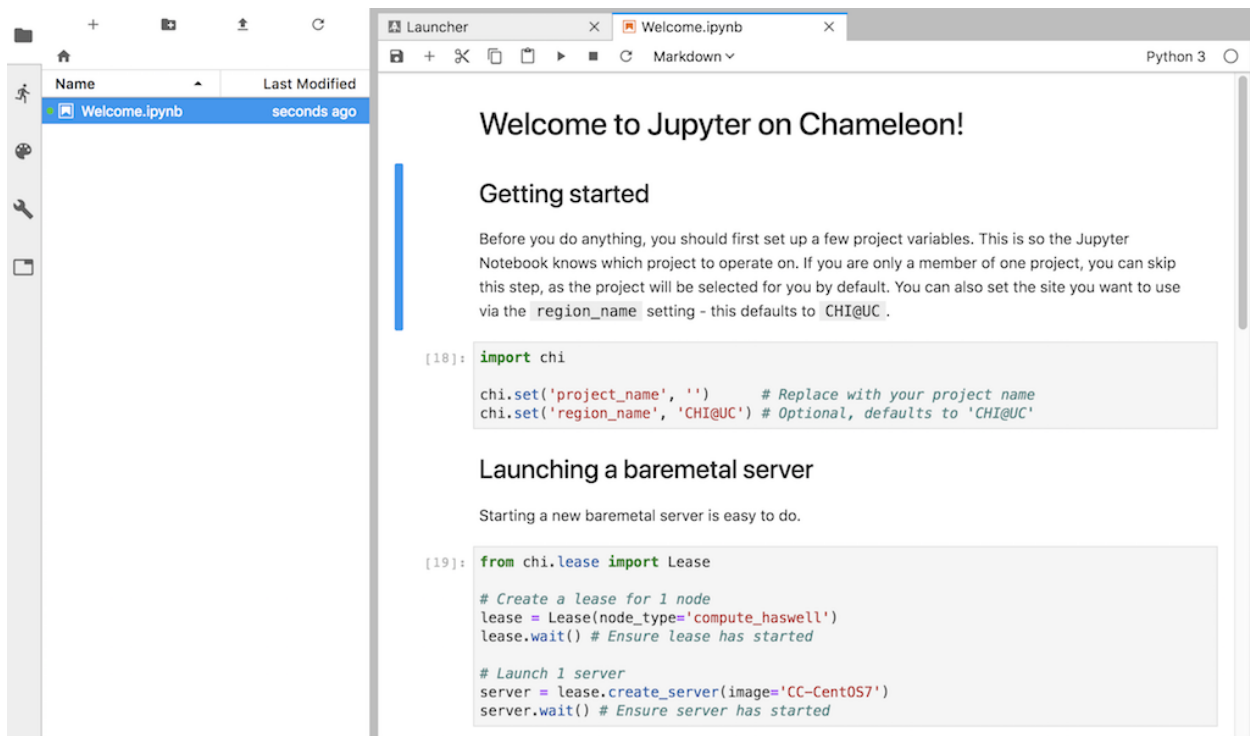
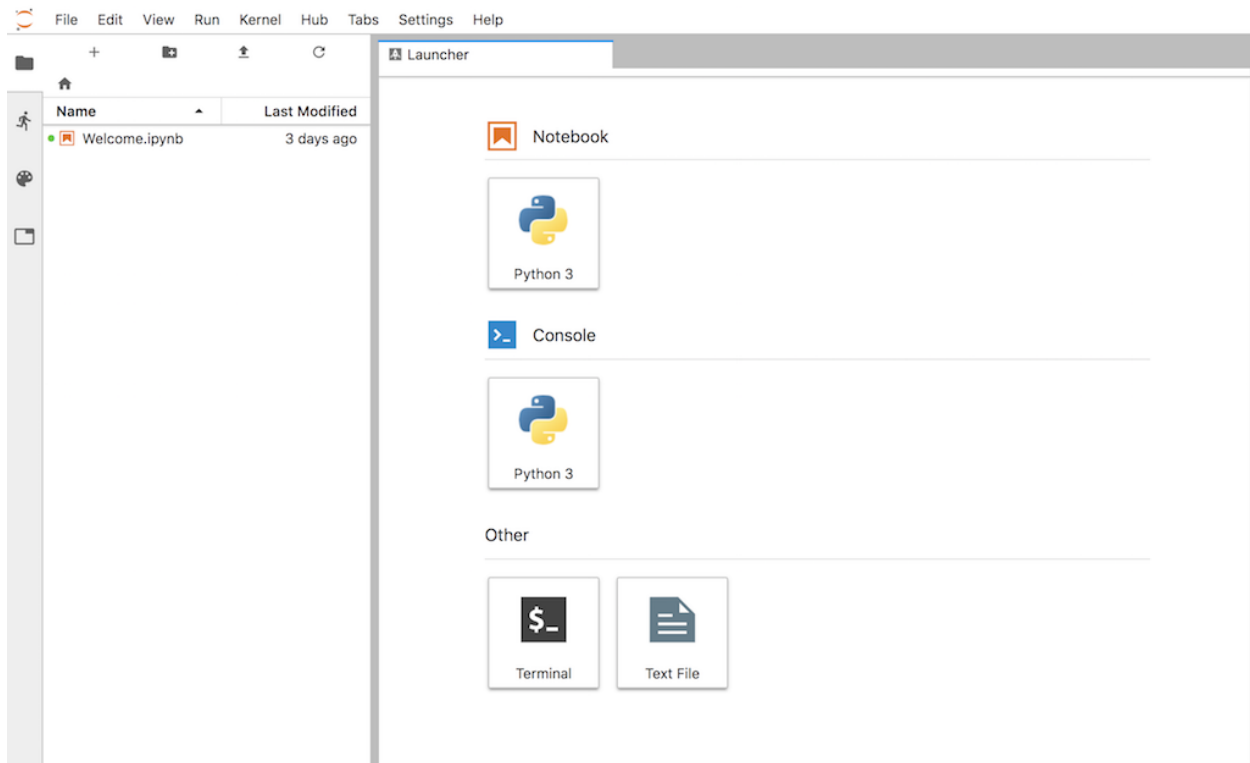
10.1 JupyterLab interface overview

When you are logged in, you will land in the JupyterLab application environment. For up-to-date documentation about the JupyterLab interface, please see the [official JupyterLab documentation](#). You will see a file browser on the left-hand side - this is your working directory. It's yours, so feel free to create and delete files as you see fit. Your working directory is initially populated with a few examples to help you get started, such as an example Notebook. Files that you save here will be persisted even if your server is torn down; the next time you log in the data will be restored. You should consider the rest of your server environment ephemeral, as updates to the Jupyter interface can cause your server to be re-built.

Hint: Jupyter Notebooks do not deal well with large files, and you should avoid trying to edit large files in the interface as it can cause instability, slowness, or even crashes. If you need to deal with large files it is best to process them on a [dedicated processing node](#), such as a baremetal node provisioned as part of your experiment.

10.2 Working with Notebooks

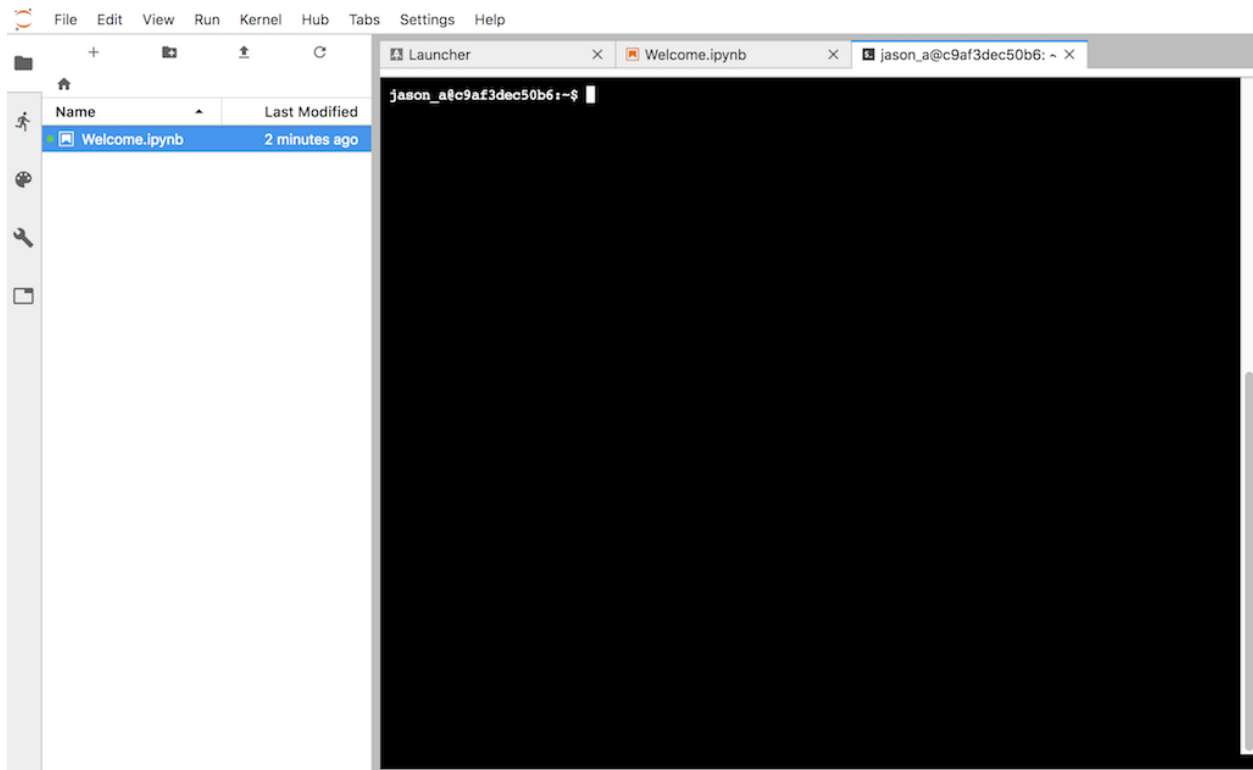
Open the “Welcome.ipynb” Notebook to see some examples of how to interface with the Chameleon testbed from within a Notebook. All Notebook servers come with OpenStack python clients installed as well as the [python-chi](#) Chameleon testbed helper library. Other python modules you may want to use in your Notebook can be installed via the [Console interface](#).



10.3 Console interface

You can open a web terminal console by going to File > New > Terminal. This works just like a remote shell, and you will also have *sudo* access so you can install additional software to support your needs.

Hint: All Chameleon Notebook servers are built from a common base image. This means if your server is torn down (which can happen during an upgrade of the Jupyter server), you may have to re-do any changes to the underlying system you made since the server was created. For this reason it is a good idea to put this setup code in a script in your working directory. Your working directory is backed up and will persist across Jupyter server restarts.



10.4 Advanced topics

10.4.1 Dedicated Jupyter Servers

The *default Jupyter environment* available to all Chameleon users is a bit limited: you are working within a shared environment and as such there are some practical limitations around the amount of CPU cores and memory you can utilize. More intensive analytical workflows may function better from within a dedicated Jupyter server for use by you and/or other members of your project.

Using the Appliance Catalog

The Chameleon *Appliance Catalog* provides a [JupyterHub appliance](#) that is functionally equivalent to the shared Jupyter environment. The appliance will allow you to reserve a Chameleon bare metal node and provision it with the JupyterHub application, along with a *Floating IP Address* that allows access over the public Internet. Any Jupyter Notebook servers managed by this multi-user environment will have access to the underlying resources on whatever node you have reserved, removing the limits around CPU and memory usage.

Using Trovi

Trovi also has a [JupyterHub artifact](#) you can instantiate on Chameleon. There is no material difference between this method and the Appliance Catalog method and it can serve as a nice introduction to the Sharing Portal if you are not already familiar with it. With this method, you actually provision your own JupyterHub server *via the shared default Chameleon JupyterHub*; it's [JupyterHub all the way down!](#)

10.4.2 Collaboration Strategies

It is often desirable to share your in-progress Notebooks with peers or supervisors for feedback, perhaps before publishing in *Trovi sharing portal*. This can be accomplished in many different ways, each suiting different use-cases. We have identified a few current tools that offer the best range of functionality.

Trovi sharing portal

The easiest way to share and collaborate on a Notebook is to publish it to *Trovi*.

Pros

- Already integrated into Jupyter; no need to sign up or log in to anything else.
- No need to download and copy Notebooks and other data around.
- Supports sharing with other Chameleon users and projects.

Cons

- Limited support for real-time collaboration; last edit wins.
- No support (yet!) for sharing one-time or expiring links with collaborators outside of Chameleon.

Google Colaboratory

Google provides a free Jupyter Notebook execution environment that can run your Notebook files in a private VM on Google's cloud infrastructure. As it is a Google product, a Google account is required to use it. Notebooks can be edited by users concurrently, similar to functionality present in Google Docs. Notebooks are stored in Google Drive and as such can be easily shared using the existing Drive sharing mechanisms. Finally, and notably, hardware-accelerated computation via GPUs and TPUs is available for free exploration. For more details see the [FAQ](#).

Pros

- Supports rich real-time collaboration on Notebook files.
- Notebooks easily sharable via Google Drive to others with Google accounts.
- Can manage access to private Notebooks via ACLs.
- Free to use.

Cons

- Not intended for long-running tasks. Your experiment may be terminated prematurely if it is deemed an invalid use of resources.
- Chameleon libraries not pre-installed. You can however install the Python API client to your Notebook via the special `!pip install python-chi` syntax. See the [Importing Libraries](#) example notebook for examples on how to install new libraries.
- Requires Google account.

GitHub + Nbviewer

A common pattern that works for many use-cases is using GitHub as the backing store for your Notebooks. This is nice because you get version history for free due to Git VCS being used behind the scenes. GitHub Notebooks are easily sharable (you just send a link) and there is decent support in GitHub for viewing the current state of the Notebook and its rendered outputs. To allow others to actually run your Notebook, you can either import the Notebook files back in to your Chameleon JupyterLab instance, or use [Binder](#), which allows spinning up a Jupyter instance for a given GitHub link.

Pros

- Supports version history via Git VCS.
- Supports easily sharing rendered Notebooks (read-only) via GitHub links.
- Can import the Notebook into a personal Jupyter server (such as the one provided by Chameleon) or via a hosted tool like Binder.
- Changes can be proposed using Pull Request workflows you may already be familiar with.

Cons

- Running the Notebook requires getting it into a Jupyter server somehow.
- Requires GitHub account if you want to keep your Notebooks private.
- Services like Binder don't create Jupyter servers with Chameleon tools (like the `python-chi` Python API) built in by default.

OVERVIEW

This section provides in-depth knowledge for utilizing Chameleon’s advanced features.

- *Resource discovery*: Discover Chameleon bare metal resources by node type and view node information.
- *Reservations*: Reserve Chameleon resources for use in your Project.
- *Bare metal instances*: Launch and manage Instances on Chameleon bare metal resources. This is a core feature of Chameleon.
- *Images*: Create images of Instances.
- *Monitoring*: Collect, manage and view experimental data from Chameleon Instances.
- *Complex appliances*: Work with Complex Appliances, which automate the process of deploying multiple Instances with reconfigurable networking.
- *Object Store*: Store user data such as files as Objects in portable Containers.
- *Shares*: Provide file storage to an instance with the OpenStack Shared File System service.
- *Networking*: Create Isolated virtual networks within Chameleon.
- *FPGAs*: Configure and work with FPGA nodes.
- *KVM*: Use non-bare metal virtual machine resources in Chameleon’s OpenStack implementation.

RESOURCE DISCOVERY

12.1 Introduction

Chameleon supports fine-grained resource discovery for experimentation, which means that you can identify a specific node, view the node's hardware maintenance history and reserve it for repeated use.

All physical resources available in Chameleon are described in the Chameleon resource registry. The resource registry is based on the [Reference API from the Grid'5000 project](#). Users can consult the registry via the resource discovery web interface or directly via REST APIs.

Note: Some resource discovery features are available through the [Chameleon Portal](#), while others are available **only** through the REST APIs.

12.2 The Hardware Catalog on the Chameleon Portal

You may use the [Hardware](#) page at the [Chameleon Portal](#) to see the different hardware resource types available at each Chameleon site.

12.2.1 Availability

The *CHI@TACC* and *CHI@UC* buttons in the *Availability* section of the Resource Browser allow you to open the Lease Calendars at the Chameleon sites. You must login using your Chameleon account to view these lease calendars.

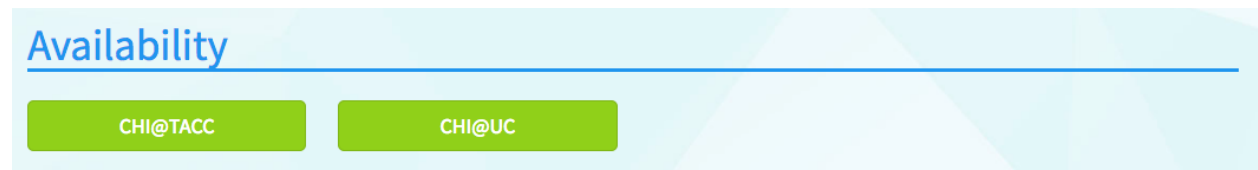


Fig. 1: Resource availability links to the lease calendars

12.2.2 Chameleon Resource Browser

The Chameleon Resource Browser allows you to filter Chameleon resources by node type and view details of each node.

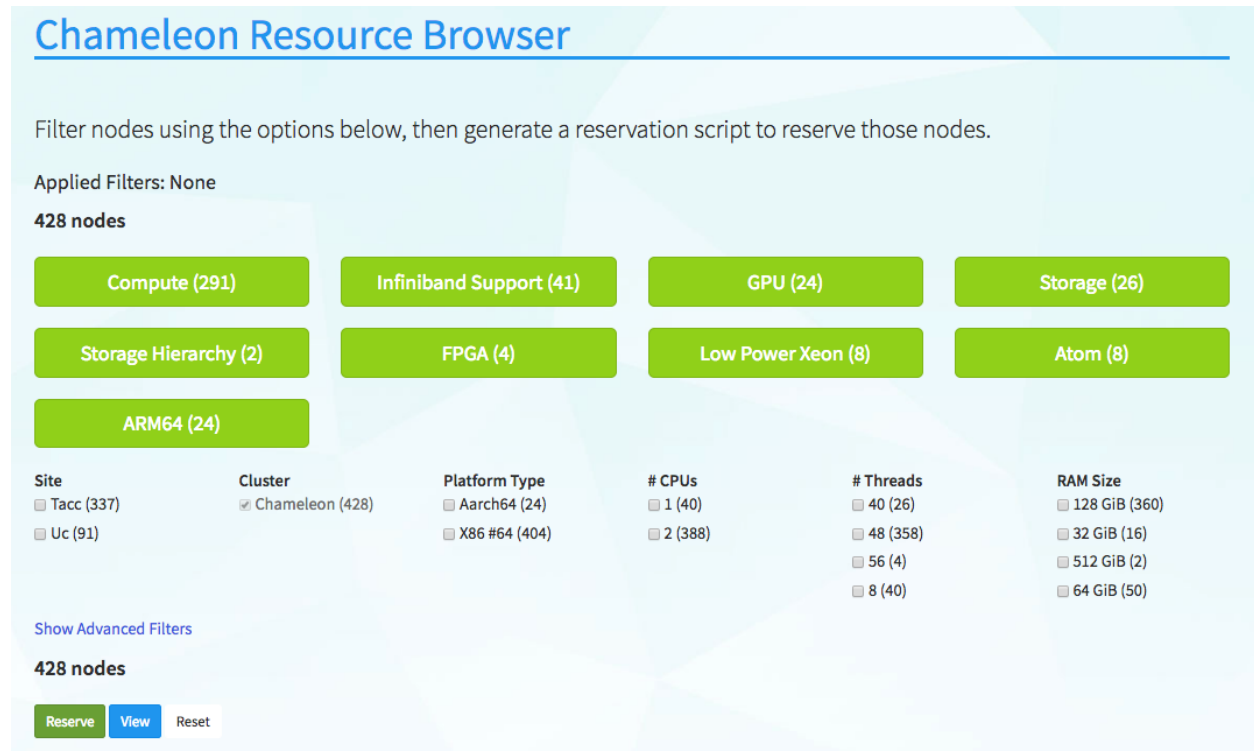


Fig. 2: The Chameleon Resource Browser

You may filter for specific node types by selecting the checkboxes that match your filter criteria or by clicking the buttons such as *Compute* and *Infiniband Support*. The numbers printed next to the node types indicate the total number of nodes of that certain type. After you have selected filter criteria, you can click the *View* button to see details of individual nodes that match your filtering criteria.

Tip: To get more precise characteristics of the selected node, search the node at [Intel's CPU database](#).

Note: All the nodes in Chameleon is identified by their *UUIDs*. You will need the *UUID* of a node for making reservations and identifying metrics collected from the node using Gnocchi. In addition, each node also has a *Version UUID*, which is used for retrieving its maintenance history.

Attention: When we replace faulty hardware on a node, the replacement part typically has the same hardware characteristics. For example, a node with a faulty 250 GB hard drive would be replaced with the same 250 GB hard drive model. However, it may be important for your experimental reproducibility to know about those hardware replacement events, in case it affects your metrics.

1. **ad89c47c-7151-44a9-9552-33056fd5547e**

Site: tacc # Threads: 48 Version: 32fd1a33731d010256cd43acbbb4469 4a61f562	Cluster: chameleon RAM Size: 137438953472	Platform Type: x86_64 Node Type: fpga	# CPUs: 2 Wattmeter: no
Bios			
Release Date: 03/09/2015	Vendor: Dell Inc.	Version: 1.2	
Chassis			
Manufacturer: Dell Inc.	Name: PowerEdge R730	Serial: 1LRKD42	
Fpga			
Board Model: 385A	Board Vendor: Nallatech	Fpga Model: Arria 10 1150 GX	Fpga Vendor: Altera
GPU			
GPU: no			
Network Adapters More ▶			
Processor			
Cache L1d: 32.00 KiB Clock Speed: 3.10 GHz	Cache L1i: 32.00 KiB Instruction Set: x86-64	Cache L2: 256.00 KiB Model: Intel Xeon	Cache L3: 30.00 MiB Other Description: Intel(R) Xeon(R) CPU E5-2670 v3 @ 2.30GHz
Vendor: Intel	Version: E5-2670 v3		
Storage Devices More ▶			
Supported Job Types			
Best Effort: no	Deploy: yes	Virtual: ivt	

Fig. 3: Node details

12.2.3 Generating a Reservation Script

The [Chameleon Portal](#) does not support a direct reservation from the [Hardware](#) page. However, you may generate a script for reserving the selected nodes by clicking on the *Reserve* button and use the auto-generated script later for the reservation.

After the form is submitted by clicking the *Generate Script* button, a new dialog that contains the auto-generated command line will show.

For node reservation using auto-generated command, please see [Provisioning and Managing Resources Using the CLI](#).

12.3 Using the REST APIs for Resource Discovery


The API is designed for users who want to programmatically discover Chameleon resources. It uses a REST architecture on top of the HTTP protocol. As a consequence, any HTTP client can be used to query the API: command-line tools (cURL), browsers, and the numerous HTTP libraries available in your favorite programming language.

It also implements the concept of “Hypermedia as the Engine of Application State” (HATEOAS), by specifying a set of hyperlinks in all responses returned by the API, which allow a user agent to discover at runtime the set of available resources as well as their semantics and content types, and transition from one resource to another.

Reservation

Start Date: *

2018-03-09



Start Time: *

09


:

00

AM

End Date: *

2018-03-09



End Time: *

11

:

00

AM

Min. Nodes Requested *

1

Max. Nodes Requested *

1

Generate Script

Cancel

Fig. 4: Generating a reservation script

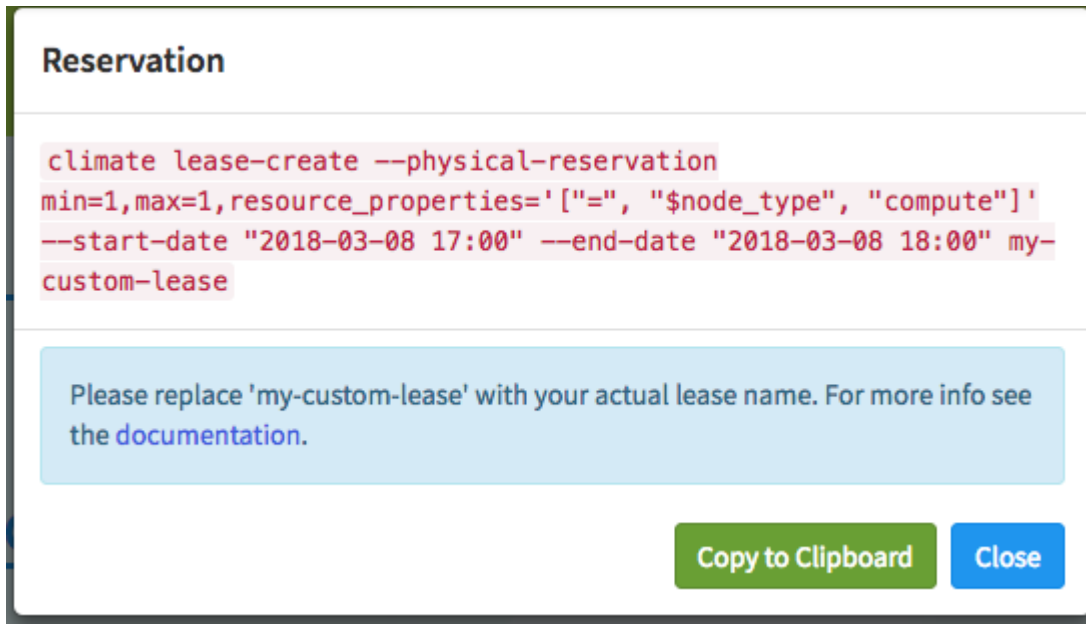


Fig. 5: An auto-generated reservation script

12.3.1 Prerequisites

Chameleon uses [cURL](#) to interact with the API. The User-Agent [cURL](#) is a command line tool for transferring data with URL syntax, supporting many protocols including HTTP and HTTPS.

To install [cURL](#), follow the instructions below:

OS X

cURL is installed by default on OS X. Nothing to do for you!

Linux

Use your package manager to install cURL. Either (Debian/Ubuntu-based distributions):

```
$ sudo apt-get install curl
```

or (RedHat-based distributions):

```
$ sudo yum install curl
```

Windows

Download and install the cURL package from [the website](#).

12.3.2 Your First Requests

The API entry-point for the resource discovery API is located at <https://api.chameleoncloud.org/>. Open your Terminal program (or the cURL executable if you're on Windows), and use cURL to fetch the resource located at that URL:

```
curl -i https://api.chameleoncloud.org/
```

Tip: The `-i` flag tells cURL to display the HTTP header in addition to the HTTP body.

Below is what you should see in response:

```
HTTP/1.1 200 OK
Server: nginx/1.6.2
Date: Thu, 19 Apr 2018 14:34:01 GMT
Content-Type: application/vnd.grid5000.item+json; charset=utf-8
Content-Length: 757
Connection: keep-alive
Allow: GET
Vary: accept
Last-Modified: Wed, 14 Mar 2018 15:05:58 GMT
ETag: "cc990a75afbc3aed5979c5cad2358b14"
Cache-Control: max-age=60, public, must-revalidate=true, proxy-revalidate=true, s-
  ↳ maxage=60
X-Info: Use `?pretty=yes` or add the HTTP header `X-Rack-PrettyJSON: yes` if you want.
  ↳ pretty output.
X-UA-Compatible: IE=Edge,chrome=1
X-Runtime: 0.034541

{"type":"grid","uid":"chameleoncloud","version":"ee0253a05223dd0f5b88df7f78fb988e67f7b039
  ↳ ","release":"3.5.7","timestamp":1524148441,"links":[{"rel":"sites","href":"/sites",
  ↳ "type":"application/vnd.grid5000.collection+json"}, {"rel":"self","type":"application/
  ↳ vnd.grid5000.item+json","href":"/"}, {"rel":"parent","type":"application/vnd.grid5000.
  ↳ item+json","href":"/"}, {"rel":"version","type":"application/vnd.grid5000.item+json",
  ↳ "href":"/versions/ee0253a05223dd0f5b88df7f78fb988e67f7b039"}, {"rel":"versions","type":
  ↳ "application/vnd.grid5000.collection+json","href":"/versions"}, {"rel":"users","type":
  ↳ "application/vnd.grid5000.collection+json","href":"/users"}, {"rel":"notifications",
  ↳ "type":"application/vnd.grid5000.collection+json","href":"/notifications"}]}
```

Note: The HTTP status of `200 OK` indicates that the server is able to process your request and that everything is fine.

Tip: By default the response body is not displayed in a pretty format. You must add the pretty query parameter to the end of the URI if you want the API to display it in a prettier way. `curl -i https://api.chameleoncloud.org/?pretty`

Attention: Do not use the pretty query parameter in your scripts, since it requires a bit more processing power to generate.

You may notice that the response contains a number of link elements, which advertise other resources that you can

access. For example, let's fetch the /sites resource.

```
curl https://api.chameleoncloud.org/sites?pretty
```

The response should look like:

```
{
  "total": 2,
  "offset": 0,
  "items": [
    {
      "description": "Texas Advanced Computing Center",
      "email_contact": "help@chameleoncloud.org",
      "latitude": 30.390223,
      "location": "Austin, Texas, USA",
      "longitude": -97.72563,
      "name": "TACC",
      "security_contact": "help@chameleoncloud.org",
      "sys_admin_contact": "help@chameleoncloud.org",
      "type": "site",
      "uid": "tacc",
      "user_support_contact": "help@chameleoncloud.org",
      "web": "https://www.chameleoncloud.org",
      "version": "ee0253a05223dd0f5b88df7f78fb988e67f7b039",
      "links": [
        {
          "rel": "clusters",
          "href": "/sites/tacc/clusters",
          "type": "application/vnd.grid5000.collection+json"
        },
        {
          "rel": "self",
          "type": "application/vnd.grid5000.item+json",
          "href": "/sites/tacc"
        },
        {
          "rel": "parent",
          "type": "application/vnd.grid5000.item+json",
          "href": "/"
        },
        {
          "rel": "version",
          "type": "application/vnd.grid5000.item+json",
          "href": "/sites/tacc/versions/ee0253a05223dd0f5b88df7f78fb988e67f7b039"
        },
        {
          "rel": "versions",
          "type": "application/vnd.grid5000.collection+json",
          "href": "/sites/tacc/versions"
        },
        {
          "rel": "jobs",
          "type": "application/vnd.grid5000.collection+json",
          "href": "/sites/tacc/jobs"
        }
      ]
    }
  ]
}
```

(continues on next page)

(continued from previous page)

```

    },
    {
      "rel": "deployments",
      "type": "application/vnd.grid5000.collection+json",
      "href": "/sites/tacc/deployments"
    },
    {
      "rel": "vlans",
      "type": "application/vnd.grid5000.collection+json",
      "href": "/sites/tacc/vlans"
    },
    {
      "rel": "metrics",
      "type": "application/vnd.grid5000.collection+json",
      "href": "/sites/tacc/metrics"
    },
    {
      "rel": "status",
      "type": "application/vnd.grid5000.item+json",
      "href": "/sites/tacc/status"
    }
  ]
},
{
  "description": "University of Chicago",
  "email_contact": "help@chameleoncloud.org",
  "latitude": 41.718002,
  "location": "Argonne National Laboratory, Lemont, Illinois, USA",
  "longitude": -87.982952,
  "name": "UC",
  "security_contact": "help@chameleoncloud.org",
  "sys_admin_contact": "help@chameleoncloud.org",
  "type": "site",
  "uid": "uc",
  "user_support_contact": "help@chameleoncloud.org",
  "web": "https://www.chameleoncloud.org",
  "version": "ee0253a05223dd0f5b88df7f78fb988e67f7b039",
  "links": [
    {
      "rel": "clusters",
      "href": "/sites/uc/clusters",
      "type": "application/vnd.grid5000.collection+json"
    },
    {
      "rel": "self",
      "type": "application/vnd.grid5000.item+json",
      "href": "/sites/uc"
    },
    {
      "rel": "parent",
      "type": "application/vnd.grid5000.item+json",
      "href": "/"
    }
  ]
}

```

(continues on next page)

(continued from previous page)

```

    },
    {
      "rel": "version",
      "type": "application/vnd.grid5000.item+json",
      "href": "/sites/uc/versions/ee0253a05223dd0f5b88df7f78fb988e67f7b039"
    },
    {
      "rel": "versions",
      "type": "application/vnd.grid5000.collection+json",
      "href": "/sites/uc/versions"
    },
    {
      "rel": "jobs",
      "type": "application/vnd.grid5000.collection+json",
      "href": "/sites/uc/jobs"
    },
    {
      "rel": "deployments",
      "type": "application/vnd.grid5000.collection+json",
      "href": "/sites/uc/deployments"
    },
    {
      "rel": "vlans",
      "type": "application/vnd.grid5000.collection+json",
      "href": "/sites/uc/vlans"
    },
    {
      "rel": "metrics",
      "type": "application/vnd.grid5000.collection+json",
      "href": "/sites/uc/metrics"
    },
    {
      "rel": "status",
      "type": "application/vnd.grid5000.item+json",
      "href": "/sites/uc/status"
    }
  ]
},
{
  "version": "ee0253a05223dd0f5b88df7f78fb988e67f7b039",
  "links": [
    {
      "rel": "self",
      "type": "application/vnd.grid5000.collection+json",
      "href": "/sites"
    }
  ]
}

```

12.3.3 Discover Resources

It is easy to discover resources using REST APIs when you chase down the links in the responses.

As seen in the previous section, when you fetch the API root resource, you can find the link to the collection of sites. If you look at the site description, you will find a list of links to other resources. For example, each site has a link named `clusters`. When you fetch this link, it returns the list of clusters on that site.

For example, to get clusters at *TACC*:

```
curl https://api.chameleoncloud.org/sites/tacc/clusters/?pretty
```

Again, you will find links in each cluster description. There is a link named `nodes` for each cluster, which as its name indicates, returns the list of nodes for the specific cluster.

For example, to get nodes on the *Alamo* cluster at *TACC* site:

```
curl https://api.chameleoncloud.org/sites/tacc/clusters/alamo/nodes/?pretty
```

You should get back a big collection of nodes. Each node is described in great details, so that you can programmatically find the cluster and nodes that are most suitable for your experiments.

The following command examples allow you to see that some of the nodes on the *Alamo* cluster at *TACC* have a different disk configuration:

```
curl https://api.chameleoncloud.org/sites/tacc/clusters/alamo/nodes/45f0fc6a-a21b-4461-
↪8414-ebf765143aad?pretty | grep -A 10 storage_devices
curl -s https://api.chameleoncloud.org/sites/tacc/clusters/alamo/nodes/0a5b61b2-dc1c-
↪4bee-86f7-247c9689ea88?pretty | grep -A 10 storage_devices
```

12.3.4 Fetch the Latest Changes

Let's go back to the site's description. In Chameleon, resources are added, updated, or removed over time. If you want to keep an eye on those changes, you can fetch the latest changes that occurred on a specific site:

```
curl https://api.chameleoncloud.org/sites/tacc/versions/?pretty
```

Each version listed in the response represents a change to some resources of the Chameleon testbed.

RESERVATIONS

Unlike virtual resources on a regular on-demand cloud, physical resources on Chameleon must be reserved before using them for an experiment. Once a reservation has been accepted, users are guaranteed that resources will be available at the time they chose (except in extraordinary circumstances such as hardware or platform failures), which helps to plan large scale experiments.

Chameleon resources are reserved via [Blazar](#) which provides Reservation as a Service for OpenStack.

Three types of resources can be reserved: physical bare metal hosts, network segments (VLANs), and floating IPs.

Attention: A note on lease stacking

To prevent resource hoarding and ensure fair access to specialized hardware, Chameleon discourages “lease stacking” or making multiple overlapping reservations. Review our [lease stacking policy](#) to align your reservations with community practices for efficient resource use.

13.1 Provisioning and Managing Resources Using the GUI

To make reservations of the resources, first log into the Horizon web interface - either [CHI@TACC](#) or [CHI@UC](#). Then, choose a project and configure your local timezone. For details on how to choose a project and update personalized settings, please see [Graphical User Interface \(GUI\)](#).

In the navigation sidebar, go to the *Reservations* section and click *Leases*.

13.1.1 The Lease Calendars

To discover when resources are available, You can access the lease calendars by clicking on the *Host Calendar* button for physical hosts and clicking on the *Network Calendar* button for VLANs. This will display a Gantt chart of the reservations which allows you to find when resources are available. The *Y* axis represents the different physical nodes in the system and the *X* axis represents time.

Tip: The nodes and VLANs are identified by their *UUIDs*. The colors are used to indicate different reservations, i.e. the resources that belong to the same reservation are colored the same. Hovering over the chart provides the details about the reservation. To change the display time frame, click on 1d, 1w, and 1m buttons or fill in the start and end times.

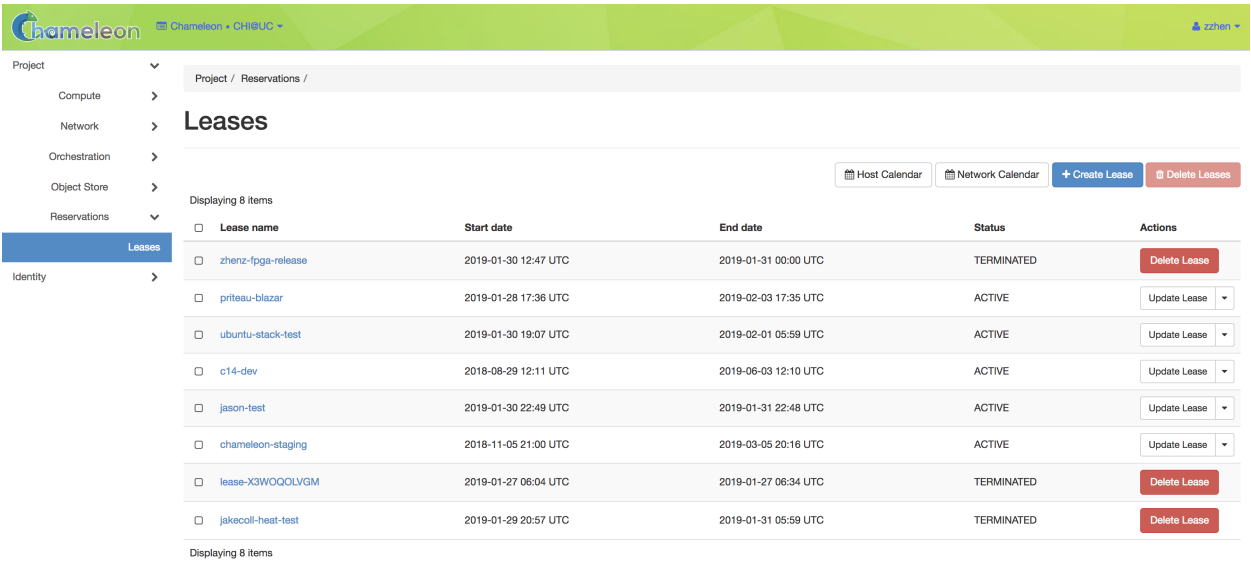


Fig. 1: The Leases page in the GUI

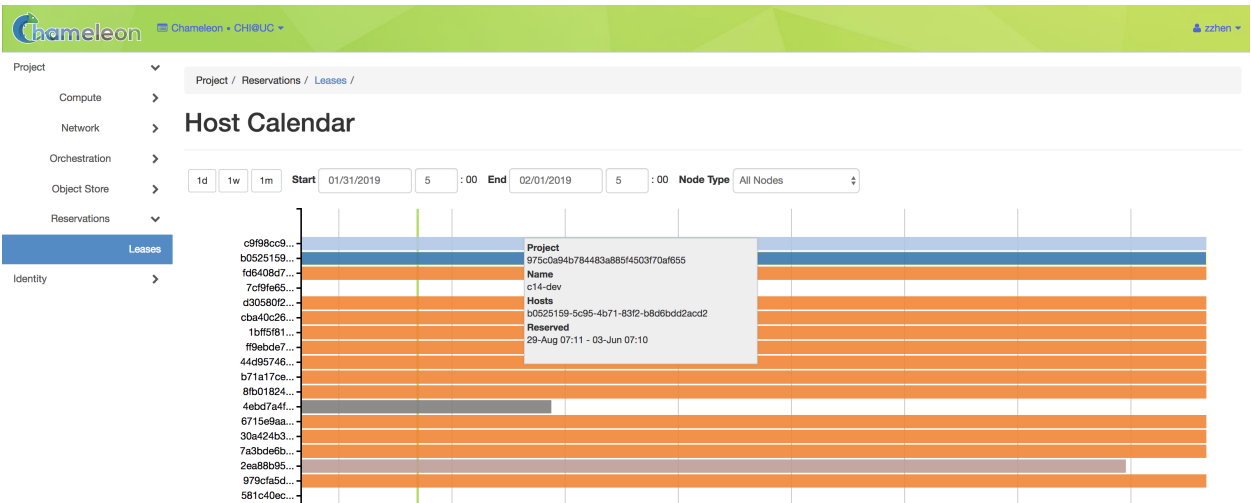


Fig. 2: The Host Calendar

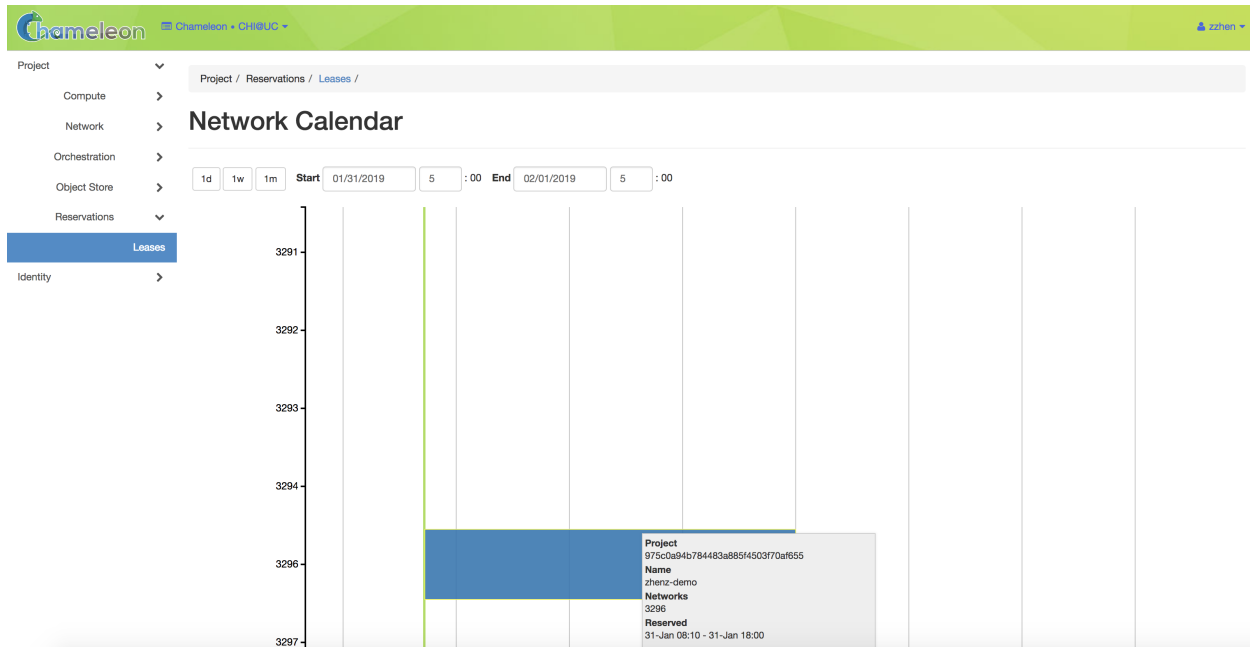


Fig. 3: The Network Calendar

13.1.2 Creating a Lease to Reserve Resources

Once you have chosen a time period when you want to reserve resources, go back to the *Leases* screen and click on the *Create Lease* button. It should bring up the window displayed below:

1. Pick a name for the lease. The name needs to be unique across your project.
2. Pick a start time and lease duration in days. If you would like to start your lease as soon as possible, you may leave the start time blank and Chameleon will attempt to reserve your nodes to begin immediately with a default Lease duration of 1 day.

Note: If you have not selected a timezone earlier, the default timezone is **UTC**. Therefore, the date must be entered in **UTC**!

Tip: You can get the UTC time by running `date -u` in your terminal.

3. To reserve a bare metal node, navigate to the “Hosts” tab.
 - a. Check “Reserve Hosts”.
 - b. Choose the minimum and maximum number of hosts.
 - c. Choose a node type in the drop down menu below the *node_type* and = drop down lists.

Note: You may only request one type of node in each individual lease. If you wish to request multiple node types, you must create separate Leases for each node type.

Create Lease

General *

Hosts

Networks

Lease Name *

Start Date ?

Today

Start Time ?

Now

Lease Length (days) ?

1

Ends ?

Tomorrow

End Time ?

Same time as now

Your timezone is currently configured as **UTC**. If you need to update your timezone please go to your [User Settings](#).

Please be courteous to other users of the testbed and make sure your lease represents a responsible use of Chameleon resources and complies with our [best practices](#). Chameleon operators reserve the right to terminate leases judged to be abusive.

For leases shorter than 24 hours, use a lease length of zero days.

Cancel

Create

Fig. 4: The Create Lease dialog

Create Lease

General *HostsNetworks

☐ Reserve Hosts

Minimum Number of Hosts ?

1

Maximum Number of Hosts ?

1

Resource Properties ?

node_type

=

compute_haswell

X

Add Filter

Cancel

« Back

Next »

For specific node reservations, you can find the node UUID using [Resource Discovery](#) on the user portal.

Fig. 5: The Create Lease dialog Host reservation tab

4. To reserve a vlan segment, navigate to the “Networks” tab.

Create Lease ✕

General * Hosts **Networks**

☐ **Reserve Network**

Network Name ?

Network Description ?

Resource Properties ?

physical_network = physnet1 ✕

☐ **Reserve Floating IPs**

Number of Floating IP Addresses Needed ?

Network name is required when reserving a network.

Fig. 6: The Create Lease dialog Network reservation tab

- a. Check “Reserve Network”
- b. Enter the network name and description

Note: When a VLAN segment reservation ends, all Neutron resources attached to the network will be automatically deleted. Bare metal instances using the network will lose network connectivity.

Tip: Network name is required when reserving VLAN segment.

5. To reserve floating IPs, navigate to the “Networks” tab.
 - a. Check “Reserve Floating IPs”.

- b. Choose the number of floating IPs.
6. Click on the *Create* button.

Once created, the lease details will be displayed. At the bottom of the page are the details about the reservation. Initially the reservation is in the **Pending** status, and stays in this state until it reaches the start time.

Tip: If you want Blazar to launch an instances or complex appliance as soon as the lease starts, read the **Advanced Reservation Orchestration** section our [Complex appliances](#) documentation.

Once the start time of the lease is reached, the lease will be started and its reservation will change to **Active**; you may need to refresh the page to see the updates.

Tip: The lease is identified by a *UUID*. You may find it useful when using the CLI or submitting tickets on our [Help Desk](#).

Attention: To ensure fairness to all users, resource reservations (leases) are limited to a duration of 7 days. However, an active lease within 48 hours of its end time can be prolonged by up to 7 days from the moment of request if resources are available.

Chameleon will send an email reminder to you 48 hours before your lease ends. If your lease duration is less than 48 hours, Chameleon will send you an email right after your lease is created. You can [disable the email notification by using the command line](#).

13.1.3 Extending a Lease

To prolong a lease, click on the *Update Lease* button in *Actions* column.

Fill out the form by specifying the amount of additional time to add to the lease. Then, click on the *Update* button to finish your request.

Tip: If there is an advance reservation blocking your lease prolongation that could potentially be moved, you can interact through the users mailing list to coordinate with others users. Additionally, if you know from the start that your lease will require longer than a week and can justify it, you can submit a ticket on our [Help Desk](#) to request a **one-time exception** of creating a longer lease.

Changing the Number of Nodes of a Lease

It is now possible to change the number of nodes reserved in a lease. For advance reservations that haven't yet started, the node count can be increased or decreased. For reservations already started, only new nodes can be added.

To change the number of nodes of a lease, click on the *Update Lease* button in *Actions* column.

Navigate to the "Hosts" tab, and fill out the form by specifying the new minimum and maximum numbers of hosts. Then, click on the *Update* button to finish your request.

Lease Detail

Lease

Name	mike-terraform
Id	6e790660-a692-473f-9429-eeeac1ae7339
Project Id	f6c7696906c04b3c89fc3bda9a1b8be0
Start date	2023-07-24 20:43 UTC
End date	2023-07-25 20:42 UTC
Status	ACTIVE
Degraded	No

Events

end_lease	<ul style="list-style-type: none">• <i>Status:</i> Undone• <i>Time:</i> 2023-07-25 20:42 UTC
before_end_lease	<ul style="list-style-type: none">• <i>Status:</i> Done• <i>Time:</i> 2023-07-24 20:43 UTC
start_lease	<ul style="list-style-type: none">• <i>Status:</i> Done• <i>Time:</i> 2023-07-24 20:43 UTC

Reservations

id	90d25f80-f25f-44b5-8168-d0767c5c0bba
status	active
resource type	physical:host
missing resources	No
resources changed	No
hypervisor_properties	-
resource_properties	-
before_end	default
on_start	default
min	1
max	1

Nodes

c01-39 (uid: 83830c1b-391b-42cb-8c2b-37d251d55ae5) (status: healthy) Re-Allocate Host

Fig. 7: Lease details page

Update Lease

General

Hosts

Lease name

Prolong for

Days

Hours

Minutes

0

0

0

Reduce by

Days

Hours

Minutes

0

0

0

Reservation values to update ?

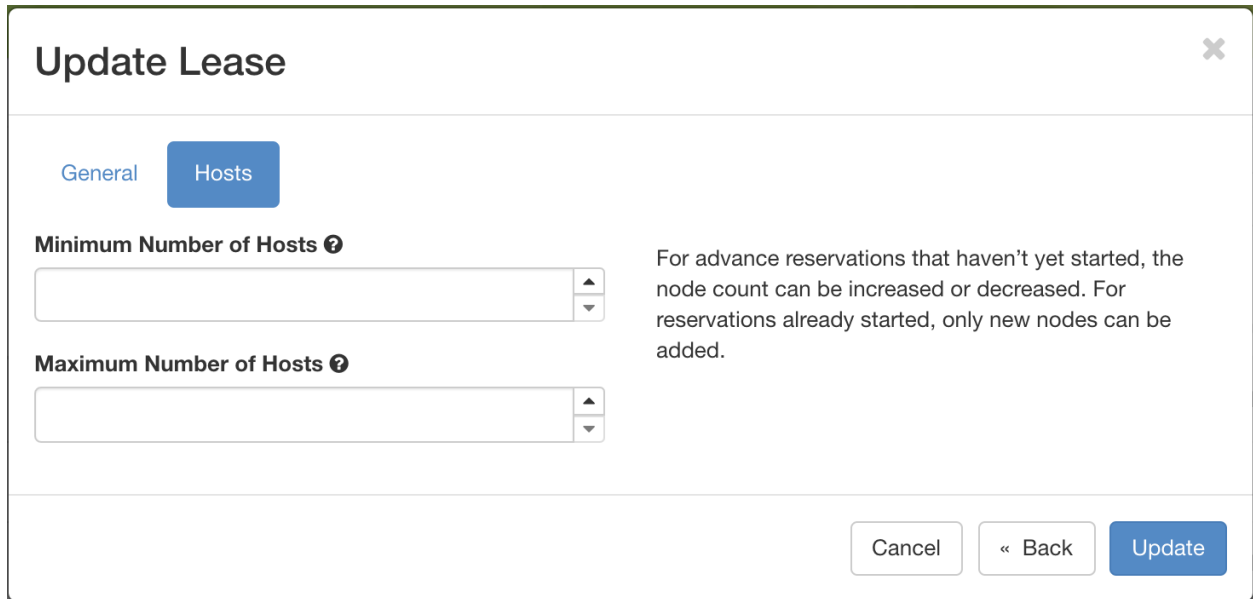
e.g.
[
 {
 "id": "087bc740-6d2d-410b-9d47-
c7b2b55a9d36",
 "max": 3
 }
]

Cancel

« Back

Next »

Fig. 8: The Update Lease Parameters dialog



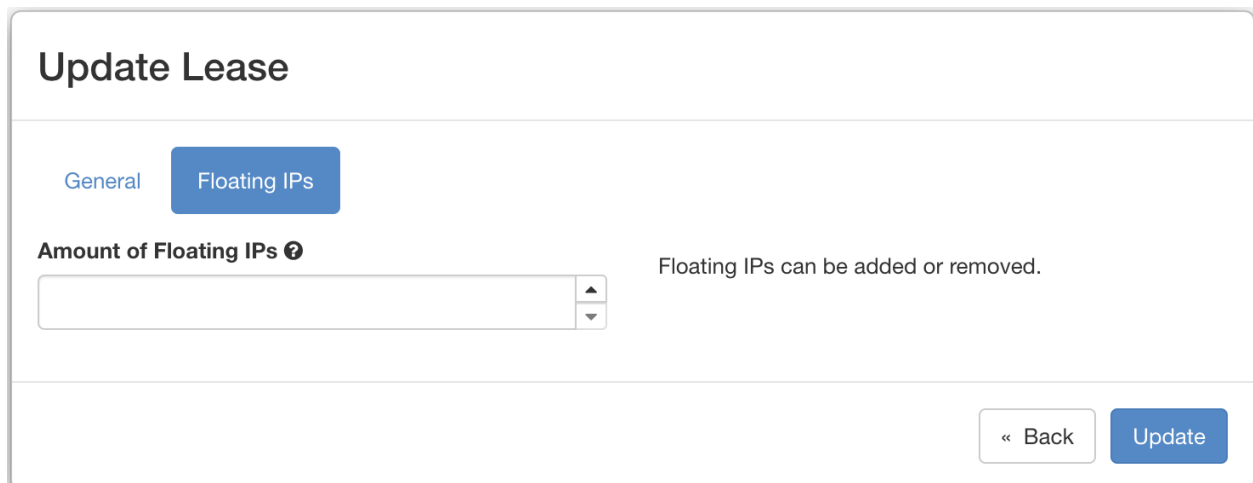
The dialog box is titled "Update Lease" with a close button (X) in the top right corner. It has two tabs: "General" and "Hosts". The "Hosts" tab is selected. Below the tabs, there are two input fields with up/down arrows. The first is labeled "Minimum Number of Hosts ?" and the second is labeled "Maximum Number of Hosts ?". To the right of these fields is a text block: "For advance reservations that haven't yet started, the node count can be increased or decreased. For reservations already started, only new nodes can be added." At the bottom right, there are three buttons: "Cancel", "« Back", and "Update".

Fig. 9: The Update Lease Parameters dialog, changing the number of reserved nodes

Changing the Number of Floating IPs in a Lease

It is possible to change the number of floating IPs in a lease, whether the lease is pending or active. In some situations, you cannot renew a lease due to another user reserving the same floating IP in your lease. In this case, you can set your lease to have 0 floating IPs, and create a second lease just for reserving floating IPs.

To change the number of floating IPs, click on the *Update Lease* button in *Actions* column.



The dialog box is titled "Update Lease" with a close button (X) in the top right corner. It has two tabs: "General" and "Floating IPs". The "Floating IPs" tab is selected. Below the tabs, there is one input field with up/down arrows labeled "Amount of Floating IPs ?". To the right of this field is a text block: "Floating IPs can be added or removed." At the bottom right, there are two buttons: "« Back" and "Update".

Fig. 10: The Update Lease Parameters dialog, changing the number of reserved IPs

Navigate to the "Floating IPs" tab, and fill out the form by specifying the amount of floating IPs. Then, click on the *Update* button to finish your request.

13.1.4 Reserving a Node by UUID

You may reserve a specific node by providing its *UUID*. To learn more about how to find a node with a specific type, please see [Resource discovery](#). In the *Create Lease* dialog, select *uid* in the *Resource Type* dropdown. Then, choose the *UUID* of the node you would like to reserve.

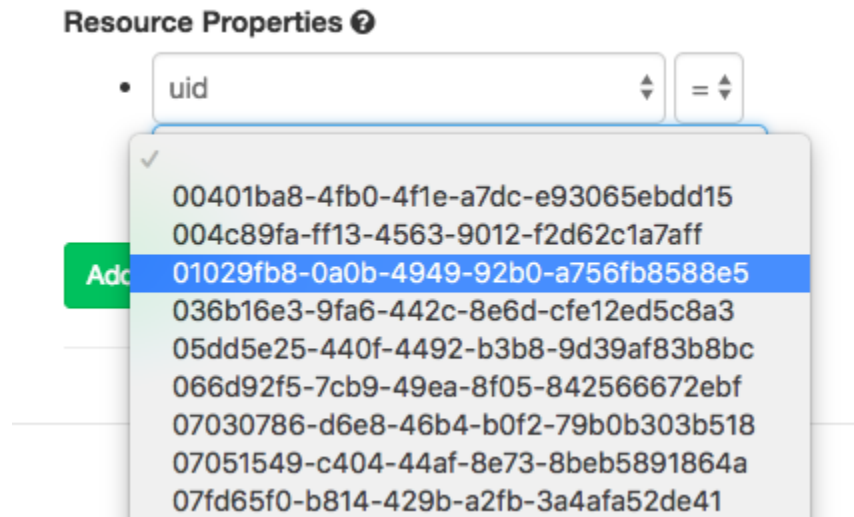


Fig. 11: Selecting a node by UUID

13.2 Provisioning and Managing Resources Using the CLI

The sections above present the most user friendly mode of usage, with most actions performed via the web interface. However, Chameleon can be accessed via the OpenStack command line tools which provides more capabilities. This section presents some advanced usage using the command line tools.

Tip: Reading *Command Line Interface (CLI)* is highly recommended before continuing on the following sections.

13.2.1 Blazar Client Installation

To reserve specific nodes, based on their identifier or their resource specifications, you must use the [Blazar](#) command line client. To use the CLI, you must install the `python-blazarclient`. To install `python-blazarclient`, run the following command:

```
pip install git+https://github.com/ChameleonCloud/python-blazarclient.git@chameleoncloud/
↪ xena
```

Note: To reserve VLAN segments or floating IPs, you must use a Chameleon fork of the Blazar client, as above.

Before using *Blazar Client*, You must configure the environment variables for your project via `source the OpenStack RC Script` or use the CLI switches every time you run the commands. Type `blazar` in your terminal session to enter the *Interactive Mode*. You may also use `blazar` in the *Shell Mode*.

13.2.2 Creating a Lease to Reserve Physical Hosts

To create a lease, use the `lease-create` command. The following arguments are required:

- `--reservation` with the `min`, `max`, `resource_type`, and `resource_properties` attributes
- `--start-date` in "YYYY-MM-DD HH:MM" format
- `--end-date` in "YYYY-MM-DD HH:MM" format
- A lease name

If `--start-date` is omitted, then the current date and time will be used by default.

For example, the following command will create a lease with the name of `my-first-lease` and the node type of `compute_skylake` that starts on June 17th, 2022 at 4:00pm and ends on June 17th, 2022 at 6:00pm:

```
openstack reservation lease create \
  --reservation min=1,max=1,resource_type=physical:host,resource_properties='["=", "
↪ $node_type", "compute_skylake"]' \
  --start-date "2022-06-17 16:00" \
  --end-date "2022-06-17 18:00" \
  my-first-lease
```

Instead of specifying the node type, you may also reserve a specific node by providing its *UUID*. For example, to reserve the node with *UUID* of `c9f98cc9-25e9-424e-8a89-002989054ec2`, you may run the command similar to the following:

```
openstack reservation lease create \
  --reservation min=1,max=1,resource_type=physical:host,resource_properties='["=", "$uid
↪ ", "c9f98cc9-25e9-424e-8a89-002989054ec2"]' \
  --start-date "2022-06-17 16:00" \
  --end-date "2022-06-17 18:00" \
  my-custom-lease
```

To create a lease with multiple resource properties, you must combine them like `["and", [property1], [property2], [...]]`. For example, to reserve a node with `$architecture.smt_size` of 48 and `node_type` of `compute_haswell`:

```
openstack reservation lease create \
  --reservation min=1,max=1,resource_type=physical:host,resource_properties='["and", ["=
↪ ", "$architecture.smt_size", "48"], ["=", "$node_type", "compute_haswell"]]' \
  --start-date "2022-06-17 16:00" \
  --end-date "2022-06-17 18:00" \
  my-custom-lease
```

Attention: To specify a `before_end` action, simply add `before_end=<action_type>` to reservation parameter. For example:

```
openstack reservation lease create \
  --reservation min=1,max=1,resource_type=physical:host,resource_properties='["=", "
↪ $uid", "c9f98cc9-25e9-424e-8a89-002989054ec2"]',before_end=email \
  --start-date "2022-06-17 16:00" \
  --end-date "2022-06-17 18:00" \
  my-custom-lease
```

Currently supported `before_end` action types include

Action Type	Description
email	Send an email notification.
default	Default action used when no action is specified; Currently set to email.
' '	Do nothing.

The default `before_end` action is set to `email`. To disable the email notification, set `before_end=' '`.

Actually, you may use any resource property that is in the resource registry to reserve the nodes. To see the list of properties of nodes, first get the full list of nodes with the command:

```
openstack reservation host list
```

The output should look like:

id	hypervisor_hostname	vcpus	memory_mb	local_gb
151	00401ba8-4fb0-4f1e-a7dc-e93065ebdd15	24	128000	200
233	004c89fa-ff13-4563-9012-f2d62c1a7aff	24	128000	200
330	01029fb8-0a0b-4949-92b0-a756fb8588e5	24	128000	200
146	036b16e3-9fa6-442c-8e6d-cfe12ed5c8a3	24	128000	200
992	05dd5e25-440f-4492-b3b8-9d39af83b8bc	8	3200	100
219	066d92f5-7cb9-49ea-8f05-842566672ebf	24	128000	200
3216	06b164d5-3514-4ebe-8928-0bd2f9508b80	0	0	0
156	07030786-d6e8-46b4-b0f2-79b0b303b518	24	128000	200
212	07051549-c404-44af-8e73-8beb5891864a	24	128000	200
175	07fd65f0-b814-429b-a2fb-3a4afa52de41	24	128000	200
255	081d2cb1-b6b5-4014-b226-7a42d8588307	24	128000	200

To get resource properties of a host, run `host-show` command with the `id` listed in the first column. For example, to get the resource properties of the host 151, run:

```
openstack reservation host show 151
```

The output should look like:

Field	Value
architecture.platform_type	x86_64
architecture.smp_size	2
architecture.smt_size	48
bios.release_date	03/09/2022
bios.vendor	Dell Inc.
bios.version	1.2
chassis.manufacturer	Dell Inc.
chassis.name	PowerEdge R630
chassis.serial	4VJGD42
cpu_info	baremetal cpu
created_at	2022-06-26 20:50:58
gpu.gpu	False
hypervisor_hostname	00401ba8-4fb0-4f1e-a7dc-e93065ebdd15

(continues on next page)

(continued from previous page)

hypervisor_type	ironic	
hypervisor_version	1	
id	151	
uid	c9f98cc9-25e9-424e-8a89-002989054ec2	
updated_at		
vcpus	48	
version	78dbf26565cf24050718674dcf322331fab8ead5	
+-----+	+-----+	+-----+

Any of the property listed in the field column may be used to reserve the nodes. For example, you can use `resource_properties='["=", "$architecture.smp_size", "2"]'` to reserve a node with two physical processors.

Note: Remember to use \$ in front of the property.

13.2.3 Extending a Lease

To extend your lease, use `lease-update` command, and provide time duration via `--prolong-for` switch. The format of the duration is a number followed by a letter specifying the time unit. `w` is for weeks, `d` is for days and `h` is for hours. For example, if you would like to extend the `my-first-lease` by one day, run the following command:

```
openstack reservation lease update --prolong-for "1d" my-first-lease
```

13.2.4 Chameleon Node Types

The following node types are reservable on Chameleon.

Node Type	<code>resource_properties='["=", "\$node_type", "<Chameleon node type name>"]'</code>
Skylake compute nodes	<code>compute_skylake</code>
Storage nodes	<code>storage</code>
Haswell Infiniband nodes	<code>compute_haswell_ib</code>
Storage Hierarchy nodes	<code>storage_hierarchy</code>
NVIDIA K80 nodes	<code>gpu_k80</code>
NVIDIA M40 nodes	<code>gpu_m40</code>
NVIDIA P100 nodes	<code>gpu_p100</code>
NVIDIA P100 NVLink nodes	<code>gpu_p100_nvlink</code>
NVIDIA RTX 6000 nodes	<code>gpu_rtx_6000</code>
FPGA nodes	<code>fpga</code>
Low power Xeon nodes	<code>lowpower_xeon</code>
Atom nodes	<code>atom</code>
ARM64 nodes	<code>arm64</code>

13.2.5 Creating a Lease to Reserve a VLAN Segment

To create a lease, use the `lease-create` command. The following arguments are required:

- `--reservation` with the `resource_type` and `network_name` attributes
- `--start-date` in "YYYY-MM-DD HH:MM" format
- `--end-date` in "YYYY-MM-DD HH:MM" format
- A lease name

Optional attributes include `network_description` and `resource_properties` which can both be added to the `--reservation` argument.

For example, the following command will create a lease with the name of `my-first-vlan-lease` and the network name `my-network` that starts on June 17th, 2022 at 4:00pm and ends on June 17th, 2022 at 6:00pm:

```
openstack reservation lease create --reservation resource_type=network,network_name="my-
↪network" --start-date "2022-06-17 16:00" --end-date "2022-06-17 18:00" my-first-vlan-
↪lease
```

Adding the `network_description` attribute provides its value as the description field when creating the Neutron network, allowing to leverage Chameleon *Software Defined Networking* features.

```
openstack reservation lease create --reservation resource_type=network,network_name="my-
↪network",network_description="OFController=${OF_CONTROLLER_IP}:${OF_CONTROLLER_PORT}" -
↪--start-date "2022-06-17 16:00" --end-date "2022-06-17 18:00" my-first-vlan-lease
```

Adding the `resource_properties` attribute allows you to reserve a specific *network segment* or *physical network* type. There are currently only two physical network types `physnet1` and `exogeni`. You can read more about both types in *Networking*. The following two examples show how to reserve a network by `segment_id` or `physical_network`.

```
openstack reservation lease create --reservation resource_type=network,network_name=my-
↪network,resource_properties='["=,"$segment_id","3501"]' --start-date "2022-06-17
↪16:00" --end-date "2022-06-17 18:00" my-first-vlan-lease
```

```
openstack reservation lease create --reservation resource_type=network,network_name=my-
↪network,resource_properties='["=,"$physical_network","physnet1"]' --start-date "2022-
↪06-17 16:00" --end-date "2022-06-17 18:00" my-first-vlan-lease
```

While separate leases can be created to reserve nodes and VLAN segments, it is also possible to combine multiple reservations within a single lease. The following example creates a lease reserving one Skylake compute node and one VLAN segment:

```
openstack reservation lease create --reservation min=1,max=1,resource_type=physical:host,
↪resource_properties='["=,"$node_type","compute_skylake"]' --reservation resource_
↪type=network,network_name="my-network" --start-date "2022-06-17 16:00" --end-date
↪"2022-06-17 18:00" my-combined-lease
```

13.2.6 Creating a Lease to Reserve Floating IPs

To create a lease, use the `lease-create` command. The following arguments are required:

- `--reservation` with the `resource_type` and `network_id` attributes
- `--start-date` in "YYYY-MM-DD HH:MM" format
- `--end-date` in "YYYY-MM-DD HH:MM" format
- A lease name

Multiple floating IPs can be reserved using the `amount` attribute. If omitted, only one floating IP is reserved.

For example, the following command will create a lease with the name of `my-first-fip-lease` that starts on June 17th, 2022 at 4:00pm and ends on June 17th, 2022 at 6:00pm and reserves three floating IPs:

```
pip install python-openstackclient
PUBLIC_NETWORK_ID=$(openstack network show public -c id -f value)
openstack reservation lease create --reservation resource_type=virtual:floatingip,
↪network_id=${PUBLIC_NETWORK_ID},amount=3 --start-date "2022-06-17 16:00" --end-date
↪"2022-06-17 18:00" my-first-fip-lease
```

13.2.7 Reallocating a Node in Your Lease

After creating your lease, you can view its details in the Horizon web interface. On this page, at the bottom, you can see a list of nodes in your lease. If you wish to reallocate one of the nodes in your lease, you can press the red “Re-Allocate Host” button next to it.

Nodes	
nc04 (uid: 1bff5f81-95b2-4d76-88b3-4a45610acb38) (status: healthy)	Re-Allocate Host
nc06 (uid: 44d95746-3573-47c2-8912-aaea639ed6ad) (status: healthy)	Re-Allocate Host
nc07 (uid: b71a17ce-fce2-4346-b943-8c49298a06db) (status: healthy)	Re-Allocate Host
nc12 (uid: 7a3bde6b-ef18-458a-9ec7-0232188d6fc3) (status: healthy)	Re-Allocate Host

Fig. 12: The nodes on the lease detail page.

You can also do the same on the command-line. Run the command that follows, entering your lease ID and the node ID where appropriate.

```
openstack reservation host reallocate --lease-id LEASE_ID NODE_ID
```

If you re-allocate a host because it is malfunctioning, please make sure to report it to the [Help Desk](#) so that we can fix it.

BARE METAL INSTANCES

Before launching an instance, make sure you own a lease. About how to create a lease, please see [Reservations](#). Once your lease is started, you are almost ready to start an instance. But first, you need to make sure that you will be able to connect to it by setting up [Key Pairs](#).

14.1 Launching instances with the GUI

14.1.1 Launch an instance

To launch an instance with the GUI, follow the steps:

1. In the navigation side bar, click *Project > Compute > Instances* to get to the *Instances* page.

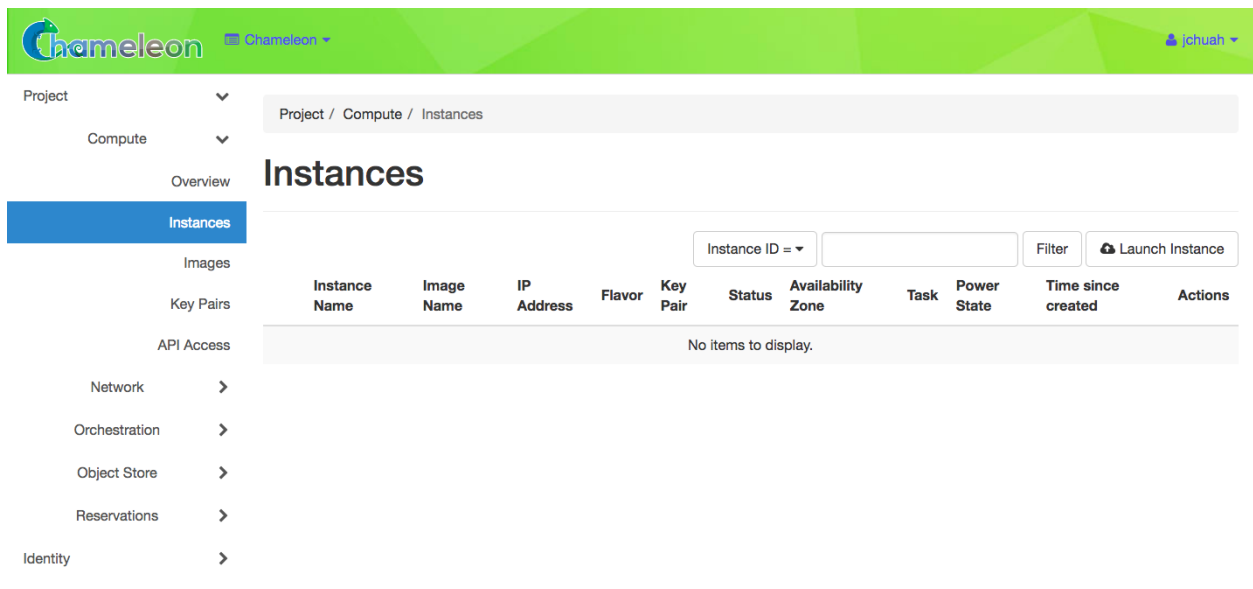


Fig. 1: The Instances page

2. Click the *Launch Instance* button in the upper right corner. This will open the *Launch Instance* wizard with several configuration steps. Steps with * are required.
3. In the *Details* step, enter a name for your instance that is unique within your project and select a currently active reservation for the instance.

Fig. 2: The Launch Instance wizard.

4. In the *Source* step, select an image for your instance and click the “up” arrow. The image should move to the *Allocated* list, and can be removed by clicking the “Down” arrow if you wish to select a different image.
5. In the *Flavor* step, select the *baremetal* flavor by clicking the “up” arrow next to it. This is the only flavor available.

Hint: If you are familiar with Openstack, other implementations allow for the selection of flavors based on machine disk size and RAM. On Chameleon, the only flavor available is “baremetal” because hardware selection is performed in reservations.

6. In the *Networks* step, select a network by clicking the “up” arrow next to it. To learn about the Chameleon default network and how to create your own network, please see [Networking](#).
7. In the *Key Pair* step, select one of your SSH key pairs. If you only have one key pair associated with your account, then it is selected by default.

Important: It is a good practice to make sure that the instance is launching with the key pair of your choice, or you will not be able to access your instance.

Tip: You may import or create key pairs directly through this step.

8. If you want to customize your instance after it has launched, you can add a customization script in the *Configuration* step.

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

Allocated

Name	Updated	Size	Type	Visibility
> CC-CentOS7	1/2/18 10:59 AM	816.30 MB	qcow2	Public

▼ Available 29

Q Click here for filters.

Name	Updated	Size	Type	Visibility
> CC-Ubuntu16.04	1/3/18 12:35 AM	668.48 MB	qcow2	Public
> CC-Ubuntu16.04-20171110	1/3/18 12:35 AM	664.49 MB	qcow2	Public
> CC-CentOS7-1710	1/2/18 10:59 AM	725.94 MB	qcow2	Public

Fig. 3: The Source configuration step

Launch Instance

Details

Source

Flavor

Networks

Network Ports

Security Groups

Key Pair

Configuration

Server Groups

Scheduler Hints

Metadata

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

Allocated

Name	VCPUS	RAM	Total Disk	Root Disk	Ephemeral Disk	Public
> baremetal	48	128 GB	128 GB	128 GB	0 GB	Yes

▼ Available 0

Q Click here for filters.

Name	VCPUS	RAM	Total Disk	Root Disk	Ephemeral Disk	Public
------	-------	-----	------------	-----------	----------------	--------

Cancel

< Back

Next >

Launch Instance

Fig. 4: The Flavor configuration step

Launch Instance

Details

Source

Flavor

Networks

Network Ports

Security Groups

Key Pair

Configuration

Server Groups

Scheduler Hints

Metadata

A key pair allows you to SSH into your newly created instance. You may select an existing key pair, import a key pair, or generate a new key pair.

+ Create Key Pair

Import Key Pair

Allocated

Displaying 1 item

Name	Fingerprint
> chameleon	d4:f8:62:c8:0d:cb:99:35:2d:d7:cc:ee:d8:77:52:c2

Displaying 1 item

▼ Available 0

Q

Click here for filters.

×

Displaying 0 items

Name	Fingerprint
No items to display.	

Displaying 0 items

×

Cancel

< Back

Next >

Launch Instance

Fig. 5: The Key Pair configuration step

- You can type in the script in *Customization Script*.
- Or you can upload your script via *Load script from a file*.

Fig. 6: Adding a Customization Script

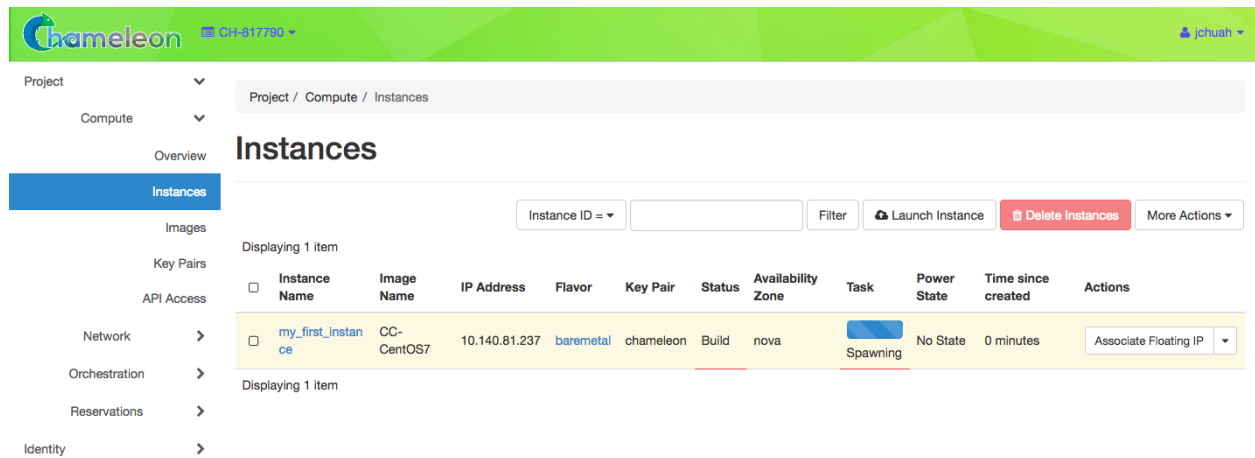
Tip: You can *disable and turn off appliance agents* using a customization script.

9. Finish configuring and start launching the instance by clicking on the *Launch Instance* button. The instance will show up in the instance list, at first in *Build* status. It takes a few minutes to deploy the instance on bare metal hardware and reboot the machine.
10. After a few minutes, the instance should become *Active*. The power state will show as *Running*. You can now *Associate a Floating IP*.
11. To view instance details, click on the instance.

14.1.2 Associate a Floating IP

To make your instance publicly accessible over the Internet, you must associate a *Floating IP Address* to it.

1. On the *Floating IPs* page (under the *Network* section in the left-hand sidebar), ensure that there is a free Floating IP available in your project. If there is not, click the *Allocate IP to Project* button to bring up the *Allocate Floating IP* dialog. In this dialog, you may simply click *Allocate IP*. You can optionally specify a description for the IP for your convenience.
2. Once a Floating IP is allocated to your project, it will display in the list view, and you can click the *Associate* button for the Floating IP to assign it to a running or spawning instance. This button will bring up the *Manage Floating IP Associations* dialog.



The screenshot shows the Chameleon Cloud web interface. The top navigation bar includes the Chameleon logo, a project ID 'CH-817780', and a user profile 'jchuah'. The left sidebar contains a menu with 'Project', 'Compute', 'Overview', 'Instances' (highlighted), 'Images', 'Key Pairs', 'API Access', 'Network', 'Orchestration', 'Reservations', and 'Identity'. The main content area is titled 'Instances' and shows a breadcrumb 'Project / Compute / Instances'. Below the title, there is a search bar for 'Instance ID', a 'Filter' button, and action buttons: 'Launch Instance', 'Delete Instances', and 'More Actions'. A table displays one instance with the following details:

Instance Name	Image Name	IP Address	Flavor	Key Pair	Status	Availability Zone	Task	Power State	Time since created	Actions
my_first_instance	CC-CentOS7	10.140.81.237	baremetal	chameleon	Build	nova	Spawning	No State	0 minutes	Associate Floating IP

Fig. 7: An Instance with the Build status



The screenshot shows the Chameleon Cloud web interface with the same layout as Figure 7. The instance 'my_first_instance' is now in the 'Active' status. The table details are as follows:

Instance Name	Image Name	IP Address	Flavor	Key Pair	Status	Availability Zone	Task	Power State	Time since created	Actions
my_first_instance	CC-CentOS7	Floating IPs: 10.140.81.237 192.5.87.96	baremetal	chameleon	Active	blazar_045a9915-4c2e-4ce7-9268-6cbe3d3b5eda	None	Running	7 minutes	Disassociate Floating IP

Fig. 8: An Instance with the Active status

Project / Compute / [Instances](#) / gnocchi_instance

gnocchi_instance

[Overview](#)[Log](#)[Console](#)[Action Log](#)

Name	gnocchi_instance
ID	b6ecb2b8-d912-46a8-b5c7-8a060e794097
Status	Build
Availability Zone	blazar_0adedd6a-ad39-4ba7-abb7-d27281a58346
Created	March 15, 2018, 2:39 p.m.
Time Since Created	0 minutes

Specs

Flavor Name	baremetal
Flavor ID	baremetal

IP Addresses

Sharednet1	10.140.81.98
-------------------	--------------

Security Groups

default	ALLOW IPv6 from default ALLOW IPv6 to ::/0 ALLOW IPv4 to 0.0.0.0/0 ALLOW IPv4 from default
----------------	---

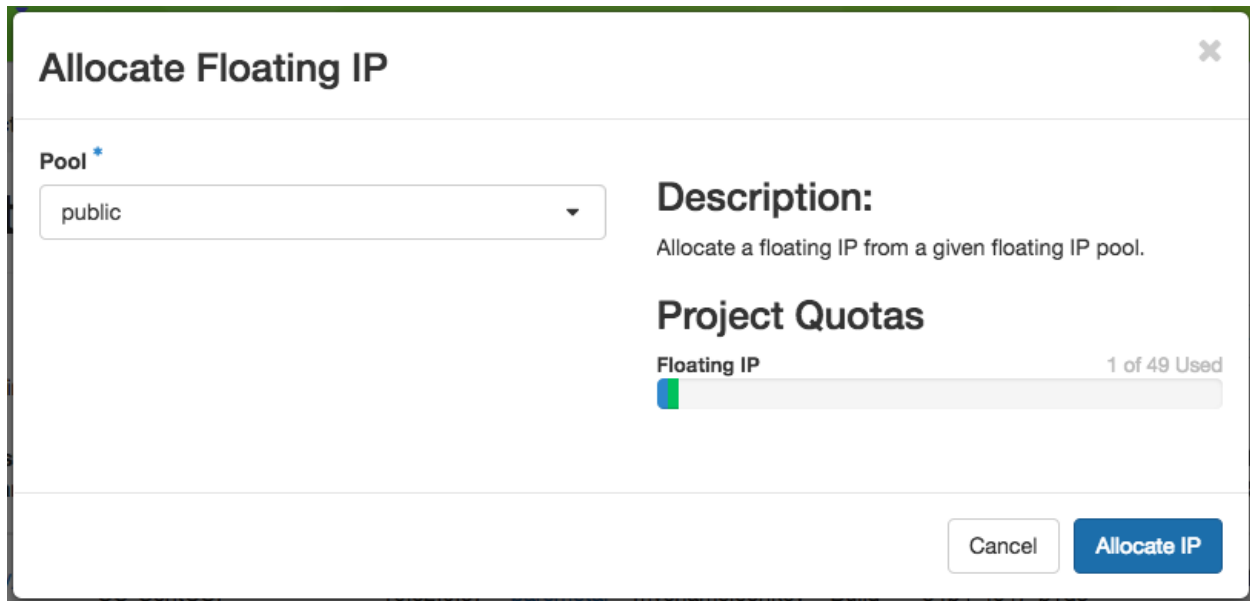
Metadata

Key Name	gnocchikey
Image Name	CC-CentOS7
Image ID	785a2bc7-4f4d-4798-8915-4c064b9f46af

Volumes Attached

Volume	No volumes attached.
---------------	----------------------

Fig. 9: Instance details



The dialog is titled "Allocate Floating IP" with a close button (X) in the top right corner. It features a "Pool" dropdown menu with "public" selected. To the right, a "Description:" section states "Allocate a floating IP from a given floating IP pool." Below this is a "Project Quotas" section with a "Floating IP" label and a progress bar showing "1 of 49 Used". At the bottom right, there are "Cancel" and "Allocate IP" buttons.

Allocate Floating IP

Pool *

public

Description:

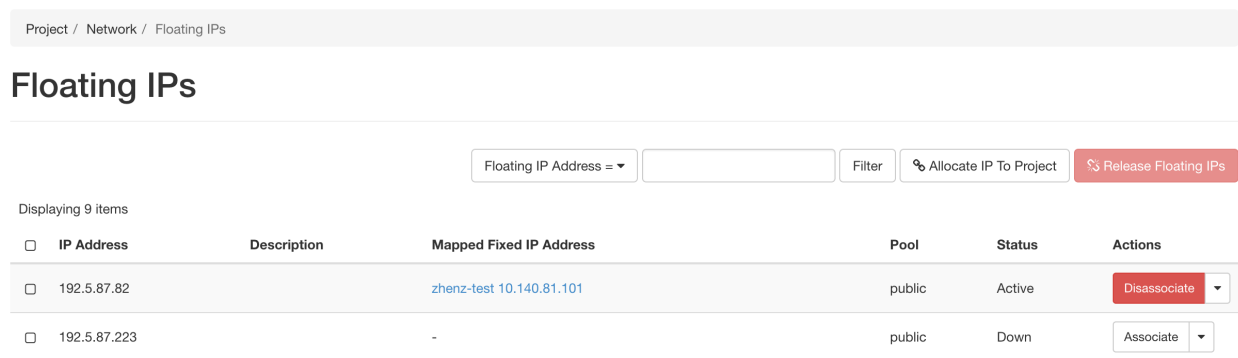
Allocate a floating IP from a given floating IP pool.

Project Quotas

Floating IP 1 of 49 Used

Cancel Allocate IP

Fig. 10: The Allocate Floating IP dialog



The interface shows a breadcrumb "Project / Network / Floating IPs" and a title "Floating IPs". Below the title are filters: "Floating IP Address" (dropdown), "Filter", "Allocate IP To Project", and "Release Floating IPs". It indicates "Displaying 9 items". The table has columns: IP Address, Description, Mapped Fixed IP Address, Pool, Status, and Actions. Two rows are shown: one with IP 192.5.87.82, description "zhenz-test", mapped IP "10.140.81.101", pool "public", status "Active", and a "Disassociate" button; and another with IP 192.5.87.223, description "-", mapped IP "-", pool "public", status "Down", and an "Associate" button.

Project / Network / Floating IPs

Floating IPs

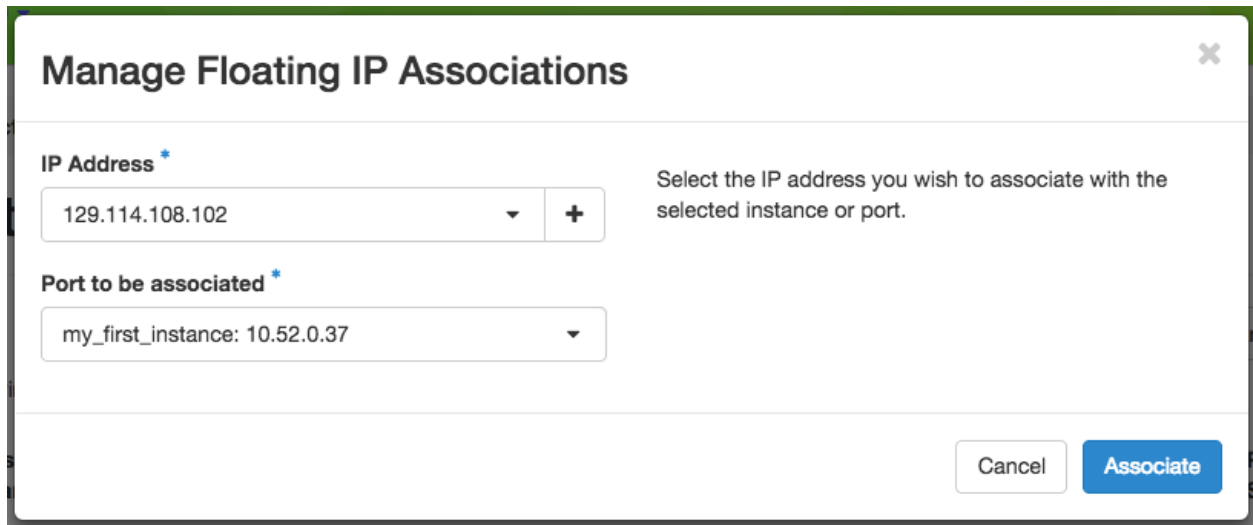
Floating IP Address Filter Allocate IP To Project Release Floating IPs

Displaying 9 items

IP Address	Description	Mapped Fixed IP Address	Pool	Status	Actions
192.5.87.82	zhenz-test	10.140.81.101	public	Active	Disassociate
192.5.87.223	-	-	public	Down	Associate

Fig. 11: The Floating IP list view with a Floating IP available

3. In the dialog, select an instance from the “Port to be associated” dropdown. Your instance’s display name will be displayed here. Click *Associate* to complete the process of assigning the IP to your instance.



Manage Floating IP Associations

IP Address *

129.114.108.102

Port to be associated *

my_first_instance: 10.52.0.37

Select the IP address you wish to associate with the selected instance or port.

Cancel Associate

Fig. 12: The Manage Floating IP Associations dialog with an IP selected

4. If you go back to the *Instances* page, you should now see the *floating IP* attached to the instance.



Chameleon OH-817780 johuah

Project / Compute / Instances

Instances

Instance ID = Filter Launch Instance Delete Instances More Actions

Displaying 1 item

Instance Name	Image Name	IP Address	Flavor	Key Pair	Status	Availability Zone	Task	Power State	Time since created	Actions
my_first_instance	CC-CentOS7	Floating IPs: 10.140.81.237 192.5.87.96	baremetal	chameleon	Active	blazar_045a9915-4c2e-4ce7-9268-6cbe3d3b5eda	None	Running	7 minutes	Disassociate Floating IP

Displaying 1 item

Fig. 13: An instance with an allocated Floating IP

14.2 Launching Instances with the CLI

Tip: Reading *Command Line Interface (CLI)* is highly recommended before continuing on the following sections.

14.2.1 Creating an instance with the CLI

To launch an instance inside a reservation, run:

```
openstack server create \  
--image CC-CentOS8 \  
--flavor baremetal \  
--key-name <key_name> \  
--nic net-id=<sharednet1_id> \  
--hint reservation=<reservation_id> \  
my-instance
```

The ID of the `sharednet1` network can be obtained using the command:

```
openstack network list
```

Alternatively, you may look it up in the GUI in the *Network > Networks* page. You can obtain your *reservation ID* via the web interface or by running:

```
openstack reservation lease show <lease_name>
```

Attention: The **reservation ID** and the **lease ID** are different

Running a script on boot

You might want to automatically execute some code after launching an instance, whether it is installing packages, changing configuration files, or running an application. OpenStack provides a mechanism called *User Data* to pass information to instances. This information can be any data in any format, but if it is a shell script it will be automatically executed after boot by `cloudinit`. You can provide this shell script either via the web interface in the *Configuration* tab when launching an instance, or by providing a file to the nova command line using the `--user-data` option.

Tip: Chameleon supported images are configured with appliance agents, including *collectd* and *Heat agents*. To turn off appliance agents on boot, in order to remove the potential impact on experimental measurements, pass the following script as `user-data`.

```
#!/bin/bash  
systemctl stop collectd.service  
systemctl disable collectd.service  
systemctl stop os-collect-config.service  
systemctl disable os-collect-config.service
```

Turning off `collectd` will **stop** collecting *Gnocchi metrics*, but you can *turn on and configure the daemon* anytime for monitoring your experiment.

14.2.2 Customizing the Kernel

It is easy to customize the operating system kernel or modify the kernel command line. You now have the option of modifying the boot loader configuration (e.g., `/boot/grub2/grub.cfg` on CentOS 7 images) to point it to a new kernel on the local disk, or specifying kernel parameters and then rebooting using this modified configuration.

To do this, you must be using a whole disk image rather than a partition image. Whole disk images contain their own kernel and ramdisk files and do not have `kernel_id` and `ramdisk_id` properties in the image repository, unlike partition images. Most Chameleon base images are whole disk images, giving you a good place to start if you're interested in custom kernels.

14.2.3 Running virtual machines on bare metal

For cloud computing and virtualization experiments, you might want to run virtual machines on bare hardware that you fully control rather than use the shared OpenStack KVM cloud. There are many different ways to configure networking for virtual machines. The configuration described below will enable you to connect your virtual machines to the Internet using a [KVM Public Bridge](#) which you must first configure manually on your host on the default network interface.

First, set up your environment for the OpenStack command line tools by following [the instructions](#). Install the [Neutron](#) client in a virtualenv with:

```
pip install python-neutronclient
```

Then, for each virtual machine you want to run, request a [Neutron](#) port with:

```
openstack port-create sharednet1
```

This should display, among other information:

- A fixed IP in the same private network as the physical nodes
- A MAC address

Finally, start your virtual machine while assigning it the *MAC address* provided by OpenStack. If your image is configured to use *DHCP*, the virtual machine should receive the allocated IP.

Neutron ports allocated this way are not automatically deleted, so please delete them after your experiment is over using:

```
openstack port delete <id>
```

You may find the ID of your ports using:

```
openstack port list
```

14.2.4 Inspecting your node

From within an instance you have already launched, you can discover which node it is running on by executing

```
curl http://169.254.169.254/openstack/latest/vendor_data.json
```

This will return a JSON dictionary describing site, cluster, and node.

14.2.5 Customizing networking

In its default configuration, the bare metal deployment system used by Chameleon ([OpenStack Ironi](#)c) is restricted to using a single shared network per site. The network configuration features available in the dashboard are not supported (Networks and Routers). On [CHI@UC](#), network layer 2 isolation is optionally available for compute nodes. You may find more details on the documentation for [Networking](#).

14.3 Interacting with instances

Once your bare metal instance has launched, you may interact with it by using SSH if you have associated a *Floating IP* to it or by using the *Serial Console* from the GUI.

14.3.1 Connecting via SSH

If you have associated a *Floating IP* with the instance and you have the private key in place, you should be able to connect to the instance via SSH using the cc account.

To access the instance using SSH, type the command in your terminal:

```
ssh cc@<floating_ip>
```

Error: If you get errors:

```
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@  WARNING: REMOTE HOST IDENTIFICATION HAS CHANGED!  @
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
IT IS POSSIBLE THAT SOMEONE IS DOING SOMETHING NASTY!
...
```

It is likely that you have saved a previous entry for the instance's *Floating IP* in your `~/.ssh/known_hosts` file on your computer. Simply removing the entry from the file should solve the issue.

You can remove the entry from the `~/.ssh/known_hosts` file by using the command:

```
ssh-keygen -R <floating_ip>
```

You may receive the response below. Type `yes` and hit enter:

```
The authenticity of host '130.202.88.241 (130.202.88.241)' can't be
↪established.
RSA key fingerprint is 5b:ca:f0:63:6f:22:c6:96:9f:c0:4a:d8:5e:dd:fd:eb.
Are you sure you want to continue connecting (yes/no)?
```

When logged in, your prompt may appear like this:

```
[cc@my-first-instance ~]$
```

Note: If you notice SSH errors such as connection refused, password requests, or failures to accept your key, it is likely that the physical node is still going through the boot process. In that case, please wait before retrying. Also make

sure that you use the cc account. If after 10 minutes you still cannot connect to the machine, please open a ticket with our [Help Desk](#).

You can now check whether the resource matches its known description in the resource registry. For this, simply run:

```
sudo cc-checks -v
```

The `cc-checks` program prints the result of each check in green if it is successful and red if it failed. You can now run your experiment directly on the machine via SSH. You can run commands with root privileges by prefixing them with `sudo`. To completely switch user and become root, use the `sudo su - root` command.

Attention: `cc-checks` is only available on legacy CentOS7 images!

14.3.2 Connecting via serial console

Chameleon now allows you to connect to the serial console of your bare metal nodes via the GUI. Once your instance is deployed, click on the *Console* button in the instance contextual menu.

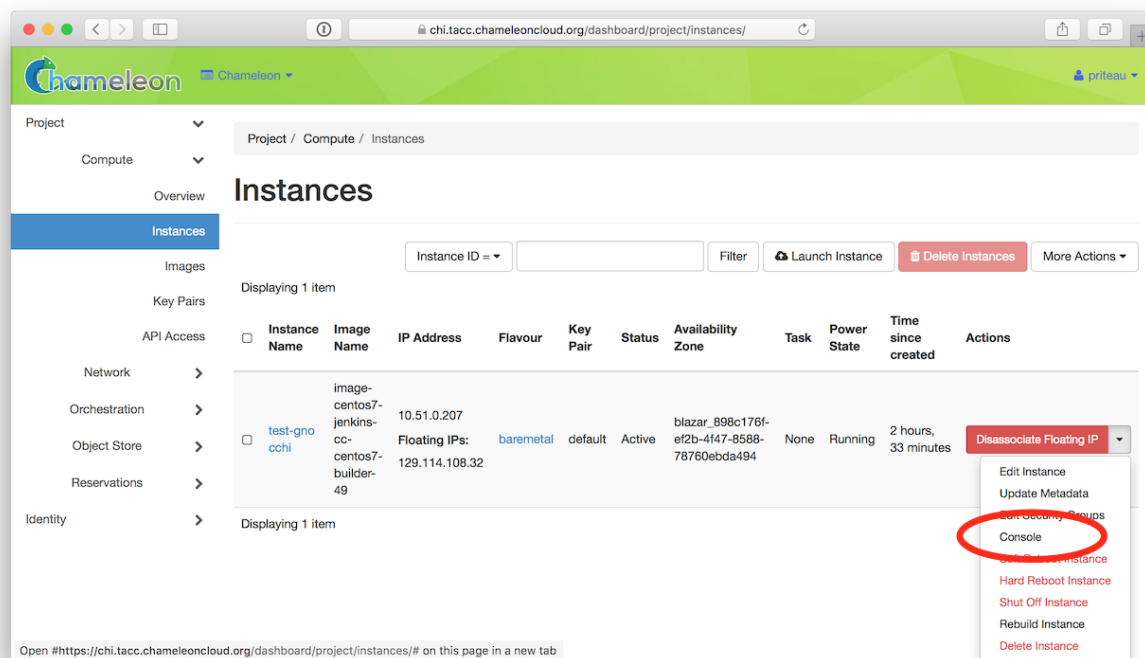


Fig. 14: The serial console button

This should open a screen showing an interactive serial console (it could take some time to show up, give it 30 seconds or so).

Our latest images are configured to auto-login into the cc account. Other images may show you a login prompt. You can set a password on the cc account by accessing it via SSH, using the command `sudo passwd cc`, and then using this password to connect to the console.

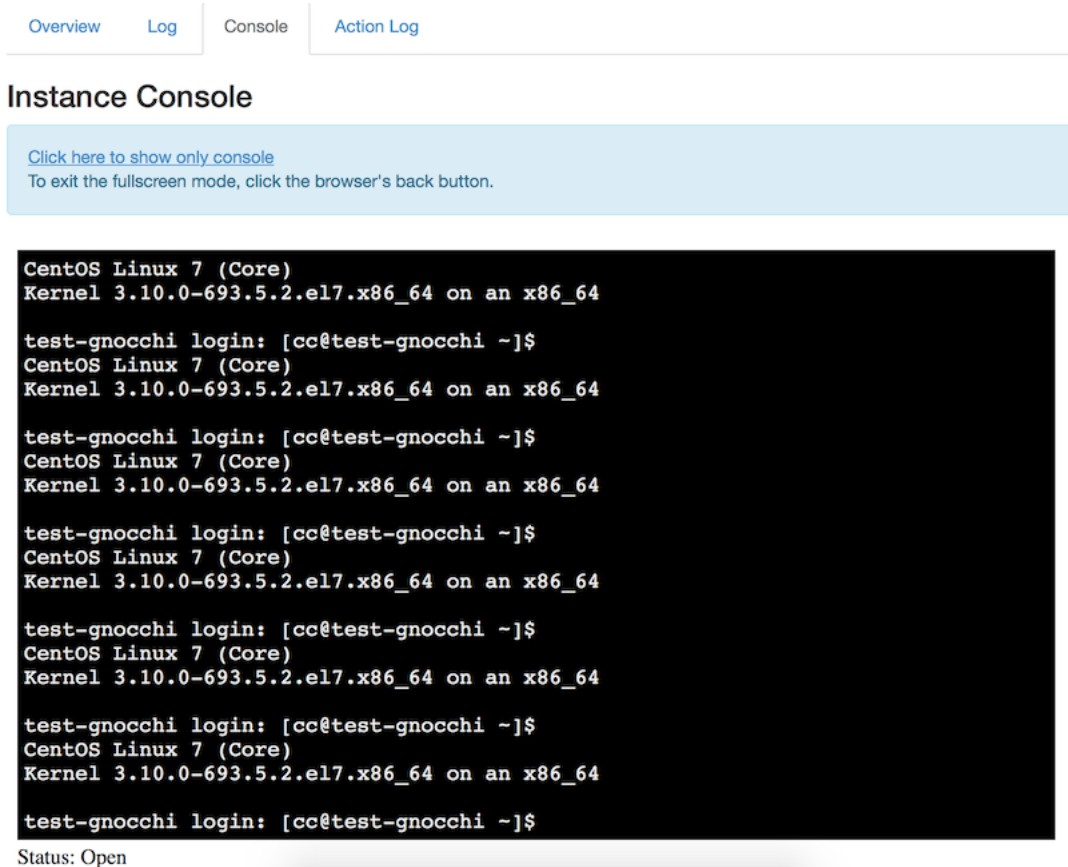


Fig. 15: An open console

IMAGES

All instances in Chameleon, whether KVM or bare metal, are running off disk images. The content of these disk images can be snapshotted at any point in time, which allows you to save your work and launch new instances from updated images later. While OpenStack KVM has built-in support for snapshotting in the Horizon web interface and via the command line, bare metal instances require a more complex process.

To work around this limitation, we provide the `cc-snapshot` utility that you can execute from inside your running instance. The `cc-snapshot` utility is pre-installed in all Chameleon supported appliances. You can find our appliances from the [Appliance Catalog](#).

The image service on Chameleon uses [OpenStack Glance](#). This documentation demonstrates how to accomplish common tasks with *Images* using the GUI and the CLI.

15.1 The `cc-snapshot` Utility

The `cc-snapshot` utility implements snapshotting a bare metal instance from command line and uploads it to [Glance](#), so that it can be immediately used to boot a new bare metal instance. The snapshot images created with this tool are whole disk images.

For ease of use, `cc-snapshot` has been installed in all the appliances supported by the Chameleon project. If you would like to use it in a different setting, it can be downloaded and installed from the [github repository](#).

To make a snapshot of a bare metal instance, run the following command from inside the instance:

```
sudo cc-snapshot <image_name>
```

Tip: You may get warnings, such as “image too large”, during snapshotting, and get prompted to confirm. If you are confident about what you are trying to do, you can skip all warnings by using the `-f` flag.

```
sudo cc-snapshot -f <image_name>
```

In addition, you can exclude directories by using the `-e` flag.

```
sudo cc-snapshot -e <dir1> -e <dir2> <image_name>
```

To see all available options for `cc-snapshot`, run `sudo cc-snapshot -h`.

You will be prompted to enter your username and password.

Tip: You can skip entering username and password by setting the `OS_USERNAME` and `OS_PASSWORD` environment variables. You can set those environment variables manually or using *The OpenStack RC Script*.

Note: When using the `cc-snapshot`, it will create an image within your project with the shared visibility. Anyone with access to your project can access this image.

Note: If you choose an *Image* name that already exists, the previous one **will not** be overwritten. A new *Image* with the same name but a different *UUID* will be generated.

Note: If you install a custom kernel, please make sure the size of your running kernel (`/lib/modules/<kernel_version>`) is less than 4GB. To find out which kernel version you're running, run `uname -r`.

Error: If you receive the following error:

```
public endpoint for image service in regionOne not found Unable to contact Glance,
↳ check username and password
```

it means that you have an outdated copy of `cc-snapshot` and you will need to update `cc-snapshot`. This usually happens when you use an older images that contains an outdated version of `cc-snapshot`.

You may also want to get new functionalities added to the latest version of `cc-snapshot`.

Run the following commands from your instance:

```
curl -O https://raw.githubusercontent.com/ChameleonCloud/cc-snapshot/master/cc-
↳ snapshot
sudo mv cc-snapshot /usr/bin/
sudo chmod +x /usr/bin/cc-snapshot
```

15.2 Managing Images using the GUI

To manage your images, use the *Images* page at [CHI@TACC](#) or [CHI@UC](#), by clicking on *Project > Compute > Images*.

Note: The Chameleon logo next to an image's name indicates that this image is an appliance supported by the Chameleon project, and is part of the Appliance Catalog.

Tip: Select *Details* from the dropdown menu to the right of any Chameleon supported appliance to view the relevant entry from the [Chameleon Appliance Catalog](#).

Note: Images at each site are stored independently. An Image made at [CHI@TACC](#) **will not** be available at [CHI@UC](#) (or vice versa) unless transferred manually.

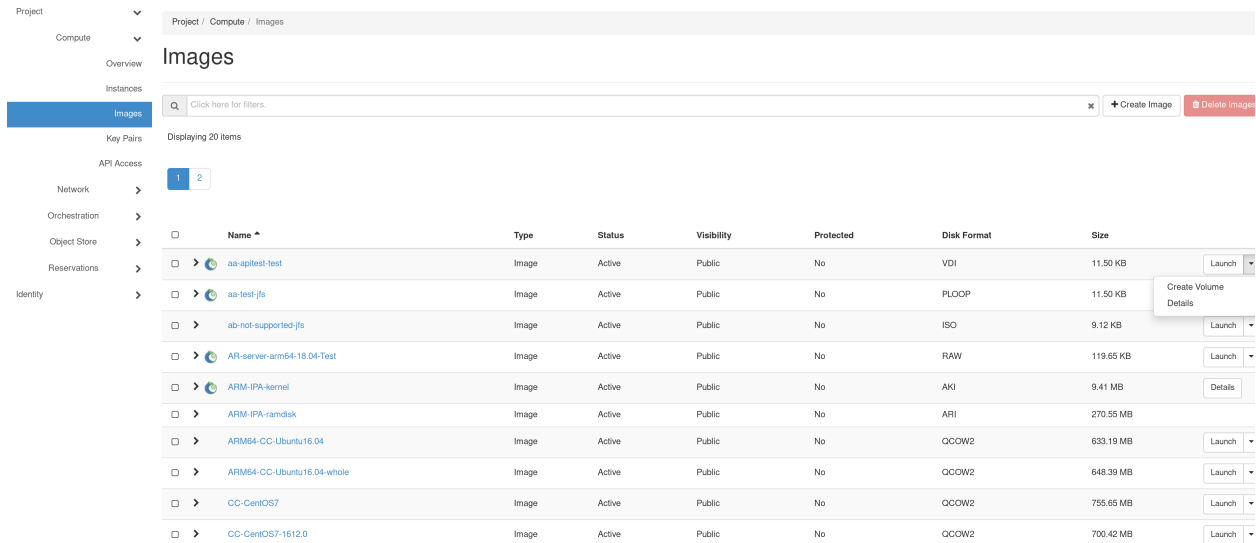


Fig. 1: The Images page

15.2.1 Uploading an Image

Use + *Create Image* button to upload an image.

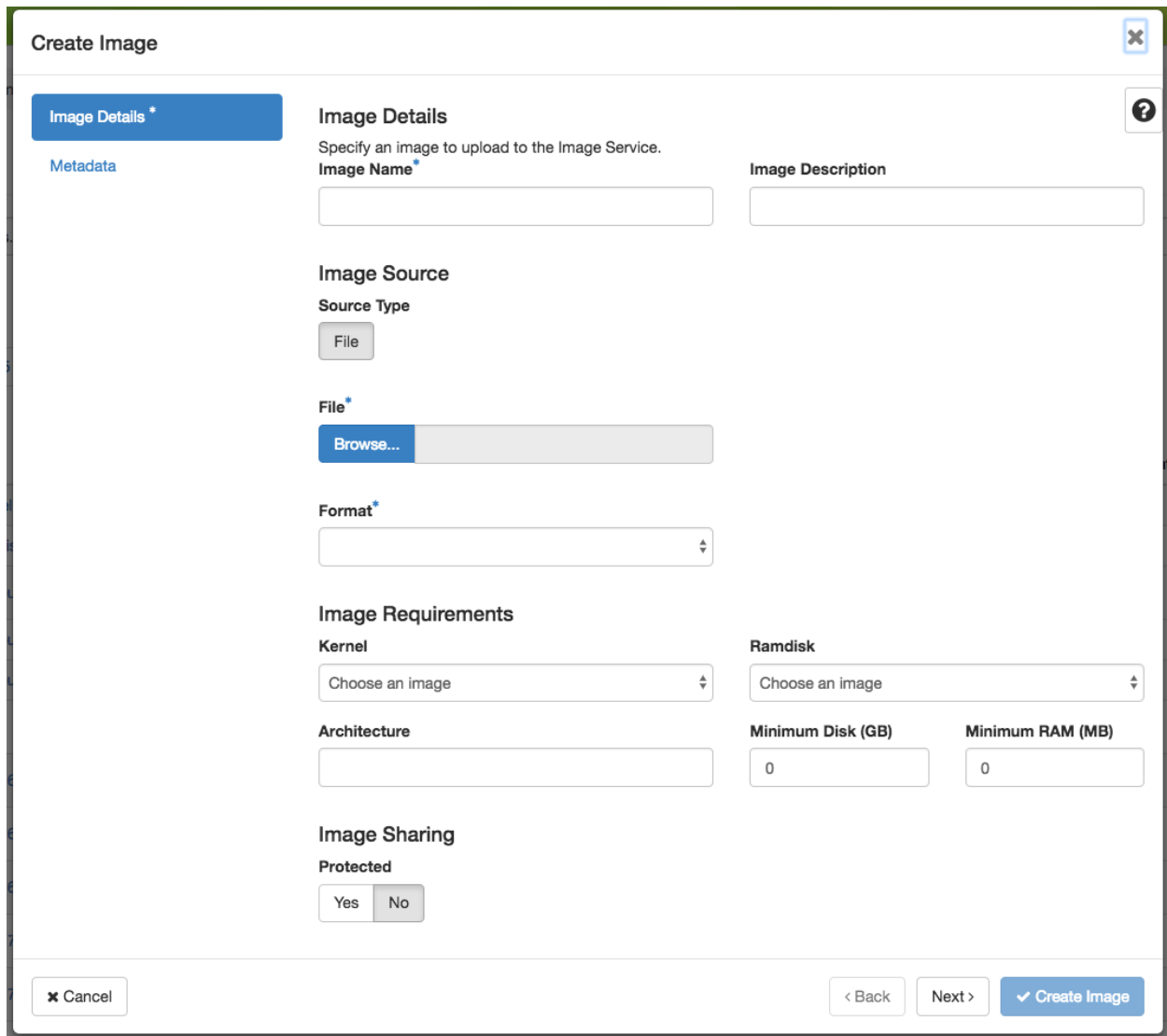
In the *Create Image* dialog:

1. Enter an *Image Name* and, optionally, a description.
2. Click *Browse* to select a file on your local machine to upload.
3. Select a *Format* of the image. Images created by the `cc-snapshot` utility are *QCOW2* images.
4. To add additional metadata for your image, use the *Metadata* section by clicking *Metadata* in the sidebar.
5. Click the *Create Image* button to upload your image.

15.2.2 Launching Instance using an Image

During the process of *launching instance* from the *Instance* page, it will ask you to select an image. Alternatively, you can launch instances with a selected image from the *Image* page by simply clicking on the *Launch* button located in the same row of the targeted image.

Tip: Other than *Launch*, there are other actions you may perform on the image. Clicking on the dropdown to explore more on what you can do.



The 'Create Image' dialog box is a window with a title bar 'Create Image' and a close button. It contains a sidebar with 'Image Details *' (selected) and 'Metadata'. The main area is divided into sections: 'Image Details' with 'Image Name*' and 'Image Description' fields; 'Image Source' with 'Source Type' (File) and 'File*' (Browse...) fields; 'Format*' (dropdown); 'Image Requirements' with 'Kernel' (Choose an image), 'Architecture' (text), 'Ramdisk' (Choose an image), 'Minimum Disk (GB)' (0), and 'Minimum RAM (MB)' (0); and 'Image Sharing' with 'Protected' (Yes/No). At the bottom are 'Cancel', 'Back', 'Next >', and 'Create Image' buttons.

Create Image

Image Details *

Image Details

Specify an image to upload to the Image Service.

Image Name*

Image Description

Image Source

Source Type

File

File*

Browse...

Format*

Image Requirements

Kernel

Choose an image

Ramdisk

Choose an image

Architecture

Minimum Disk (GB)

0

Minimum RAM (MB)

0

Image Sharing

Protected

Yes No

✕ Cancel < Back Next > ✓ Create Image

Fig. 2: The Create Image dialog

15.2.3 Viewing Image Details

To view image details, click on the name of the Image.

Project / Compute / Images

Back

wireshark-demo

Launch

Create Volume

Edit Image

Update Metadata

Delete Image

Image	Security
<div><div>ID</div><div>431bc3b8-8253-45a5-9a8b-c12d7a80e7a9</div><div>Type</div><div>Image</div><div>Status</div><div>Active</div><div>Size</div><div>1.04 GB</div><div>Min. Disk</div><div>0</div><div>Min. RAM</div><div>0</div><div>Disk Format</div><div>QCOW2</div><div>Container Format</div><div>BARE</div><div>Created At</div><div>2018-03-05T17:36:02Z</div><div>Updated At</div><div>2018-03-05T17:36:12Z</div></div>	<div><div>Owner</div><div>0283d33910314260ac449d2b5442ba43</div><div>Filename</div><div>-</div><div>Visibility</div><div>shared</div><div>Protected</div><div>No</div><div>Checksum</div><div>d54b44459c72cc29edd7dc5f445b607d</div></div>

Fig. 3: Image details

The dropdown list in the top right corner allows you to perform various actions on the selected image, such as *Launch*, *Edit Image*, and *Update Metadata*.

Tip: The *ID* on the image details' page is useful when you work on the image using the CLI.

15.2.4 Publishing Images to the Appliance Catalog

API Access

12

Network

Orchestration

Object Store

Reservations

Identity

	Name	Type	Status	Visibility	Protected	Disk Format	Size
	fnib_test_v2	Image	Active	Private	No	ISO	132.57 MB
	aa-apitest-test	Image	Active	Public	No	VDI	11.50 KB
	ab-not-supported-jfs	Image	Active	Public	No	ISO	9.12 KB
	aa-test-jfs	Image	Active	Public	No	PLOOP	11.50 KB

Create Volume

Edit Image

Update Metadata

Delete Image

Publish to Appliance Catalog

The dropdown menu to the right of listed images allows their owners to publish an appliance to the [Appliance Catalog](#). Select *Publish to Appliance Catalog*.

The *Create Appliances* web form will open automatically with most fields pre-populated. Complete the form and select *Create an Appliance*.

Entering a descriptive name, author and support contact information, the version, and an informative description can be helpful and is encouraged. **The description is used by others to determine if an appliance contains the tools needed for their research.**

Tip: To make your description effective you may want to ask the following questions:

- What does the appliance contain?
 - What are the specific packages and their versions?
 - What is it useful for?
 - Where can it be deployed and/or what restrictions/limitations does it have?
 - How should users connect to it and what accounts are enabled?
-

15.3 Managing Images using the CLI

Tip: Reading *Command Line Interface (CLI)* is highly recommended before continuing on the following sections.

15.3.1 Uploading an Image

After configuring the environment variables using *The OpenStack RC Script*, run the following command:

```
openstack image create --file <file> --disk-format <format> <image-name>
```

Provide the path to and the name of your image file in your local file system as the value of the `file` parameter. Also, indicate the image format using the `format` switch, such as QCOW2. Finally, name your image via the `image-name` switch.

15.3.2 Downloading an Image

Downloading an image file to your local machine is **only** available via the CLI. You may find it useful when transferring images from one Chameleon site to another. To download an image file, run the following command:

```
openstack image save --file <filename> <image>
```

Use `filename` to indicate where you would like to save the image in your local file system. Also, replace `image` with either the name or the *ID* of the image on Chameleon.

Important: If you do not provide the `--file` parameter, it will print out the binary image data in your terminal.

15.3.3 Retrieving Images

You may list all images of your project by typing:

```
openstack image list
```

Optionally, you may add filters to the list, such as `--shared` to only display the images shared within your project. Use `openstack image list --help` to see all the available filters.

15.3.4 Viewing Image Details

You may view details of an image with the command:

```
openstack image show <image>
```

Replace `image` with either an image name or its *UUID*.

15.3.5 Sharing an Image

You may share images several ways. If you wish to share an image with everyone, use:

```
openstack image set --public <image>
```

Replace `image` with the image *UUID*.

If you would like to share an image with another project, first set the image visibility to shared:

```
openstack image set --shared <image>
```

Next add the project you wish to share the image with:

```
openstack image add project <image> <project>
```

Replace `image` and `project` with the corresponding *UUIDs*

Finally the project that the image is shared to must accept the shared image. Run this command with a user in the second project:

```
openstack image set --accept <image>
```

Replace `image` with the image *UUID* and the second project should now be able to use the image!

Important: Only the owner of the image can modify it or any properties. However a project who has an image shared to it can remove themselves from the list of image members.

15.3.6 Editing an Image

You may edit an image using the command:

```
openstack image set <image> ...
```

Replace `image` with either an image name or its *UUID*. You must provide additional flags to update an image. Use `openstack image set --help` to see all the options.

MONITORING

Warning: Gnocchi is not supported on the testbed. If you are interested in power monitoring, you should see [this trovi artifact](#)

Chameleon collects monitoring information, representing qualities such as CPU load or power consumption data, from various sources into an *aggregation service*. Data is kept in this service with resolution that decreases over time. Users can retrieve those metrics via a *command line interface (CLI)*.

In Chameleon, the aggregation service is implemented using the [Gnocchi time series database](#). All Chameleon supported images, from which most of our user's images are derived, are configured to send a selection of system metrics using the [collectd system statistics collection daemon](#). There is a wide range of qualities this daemon can gather; by default only selected metrics are sent but users can configure the daemon (see [Configuring collectd](#)) to adapt this set anytime to monitor their experiments better. Another source of metrics is the infrastructure itself, for example the energy and power consumption metrics.

Tip: Reading *Command Line Interface (CLI)* is highly recommended before continuing on the following sections.

16.1 Setting up the Gnocchi CLI

In addition to *Installing the CLI*, you must also install the Gnocchi client plugin. To install on your local machine, run the following command:

```
pip install gnocchiclient
```

Then, set up your environment for OpenStack command line usage, as described in *The OpenStack RC Script*.

16.2 Retrieving Metrics

Now, you can run the `gnocchi` command line utility. To show the different kinds of metrics collected for a specific instance, run:

```
gnocchi resource show <instance_id>
```

Tip: You can get the instance' *ID* from the GUI.

You can get your list of instances by running:

```
gnocchi resource list
```

It will print out a chart similar to below:

id	type	project_id
8d643431-9a90-4100-8e00-f43d56a68d1e	generic	None
39ff85e4-cf4e-4969-9408-af47a372ad06	generic	None
3c6c81ba-0566-4cde-a8c5-7ae4d4644293	generic	None
219f2fec-0e90-4e04-a5d7-1a78c9fde93b	generic	None
57f2ba05-e57c-4241-bd27-bf95cca9c027	generic	None
a0cc7bb7-0169-4973-8d4a-08151c52dec6	generic	None
afb1d1e2-85db-463c-9769-2a2752eb447e	generic	None
87e52c8d-c66e-43f5-b9fc-da376eccdf2d	generic	None
bf383c17-d76a-4e50-b347-426c96020d3b	generic	None
9f25dfffd-79f5-4c34-86b6-79767b8db086	generic	None
4b8ee1ce-9733-4808-921f-6d8ca92a6752	generic	None
5887a427-286f-47ad-bd4a-d7b9278bbc0f	generic	None
f5856741-89d5-462f-a0a2-f2423d9bfc38	generic	None
fea54e18-9668-4df0-a511-5b2af4c76945	generic	None
304dc702-c57a-471c-81df-6e711d793e50	generic	None

You will get a result like the following:

Field	Value
created_by_project_id	2c8f25efb722467eb9fc25f38996b7c4
created_by_user_id	7961a8c338ba4cb8a4ac6dfe0ab333f5
creator	7961a8c338ba4cb8a4ac6dfe0ab333f5:2c8f25efb722467eb9fc25f38996b7c4
ended_at	None
id	304dc702-c57a-471c-81df-6e711d793e50
metrics	interface-en0@if_dropped: 511abf80-d9e9-4e37-bde6-b34de19a7a87
	interface-en0@if_errors: 7bf316e3-ce63-424c-955c-1654541dafea
	interface-en0@if_octets: 0b9a204b-38fd-4b4f-a5a1-c25b9b739c5c
	interface-en0@if_packets: a62006be-d45a-4b2c-a201-4f1b4770f43c
	interface-eno2@if_dropped: 56bd5603-ed8c-401c-891e-05170e3b40a7

(continues on next page)

(continued from previous page)

	interface-eno2@if_errors: 5d2d1a60-1ca8-4256-a395-0125428cf395	└─
↪		
	interface-eno2@if_octets: 3f3daf4b-2ef8-4383-b031-294e51487ae9	└─
↪		
	interface-eno2@if_packets: 0aa3fb64-764f-402b-b9eb-	
↪ 6fb47e3d0efc		
	interface-eno3@if_dropped: 23c59f0f-d018-4538-a387-	
↪ 90bd5809a0f0		
	interface-eno3@if_errors: c8ab32bb-02e7-48f7-8a67-92cf96aa6974	└─
↪		
	interface-eno3@if_octets: be37ef63-9ed5-4547-851e-46f1aa2e91d6	└─
↪		
	interface-eno3@if_packets: 149ae533-2f03-4a87-91a6-	
↪ 6aa0f8a541b3		
	interface-eno4@if_dropped: 6b8285d5-7e87-4f10-8abc-	
↪ 1ac848bf8240		
	interface-eno4@if_errors: 0dcd9925-c6e6-480d-88cb-6eb099cd4650	└─
↪		
	interface-eno4@if_octets: 4ff866fd-d5ef-4a55-aeab-7cfbe1ac1f28	└─
↪		
	interface-eno4@if_packets: 0fe10bf7-79ab-4bfb-aa6b-	
↪ 64efd3b925c1		
	interface-lo@if_dropped: 39318dc7-f008-4258-8832-457c90193924	└─
↪		
	interface-lo@if_errors: f3998907-786f-4ffd-a47b-bea1f4b9ad97	└─
↪		
	interface-lo@if_octets: f01791f8-8939-4bf3-aae7-abb1e4bffc2e	└─
↪		
	interface-lo@if_packets: 6aaf06ee-5a8d-49f2-b7b9-c1d27841a89b	└─
↪		
	load@load: 8d6132f8-6e60-409b-8d64-7092491aa9db	└─
↪		
	memory@memory.buffered: a6ad6e20-f951-4152-aac3-d6d081c33c09	└─
↪		
	memory@memory.cached: ca0e3b30-b450-484b-ac41-a03424da279b	└─
↪		
	memory@memory.free: 7aee53a8-93f9-4bac-92e3-7694b219c698	└─
↪		
	memory@memory.slab_recl: 074897b8-c40e-4538-9ef6-69338764bed3	└─
↪		
	memory@memory.slab_unrecl: 1bb6c19d-e788-40cd-98f0-	
↪ 0c5820e03563		
	memory@memory.used: 8b56e1ea-0aaa-4c1b-9462-f3698bad2ca7	└─
↪		
original_resource_id	304dc702-c57a-471c-81df-6e711d793e50	└─
↪		
project_id	None	└─
↪		
revision_end	None	└─
↪		
revision_start	2018-02-15T15:42:18.495824+00:00	└─
↪		

(continues on next page)

(continued from previous page)

```
| started_at          | 2018-02-15T15:42:18.495781+00:00  
↩ |  
| type               | generic  
↩ |  
| user_id            | None  
↩ |  
+-----+-----+
```

To get all the measurements of a particular metric, run:

```
gnocchi measures show <metric_name> --resource-id <instance_id> --refresh
```

For example, to get measurements of used memory over time for instance d17d5191-af60-4407-9ed2-e3d48e86ac6d, run:

```
gnocchi measures show memory@memory.used --resource-id d17d5191-af60-4407-9ed2-  
e3d48e86ac6d --refresh
```

Tip: You may notice that each metric has been assigned a *UUID* to it. Therefore, instead of providing `metric name`, you can provide `metric uuid`.

This will show the latest measurements of that metric with granularity set to 1.0, as well as aggregate values (by default, the mean) over one minute and one hour. Other aggregation methods can be used with the `--aggregation` option, such as `std`, `count`, `min`, `max` and `sum`. Your results may appear like this:

timestamp	granularity	value
2017-12-22T18:00:00+01:00	3600.0	1222193280.0
2017-12-22T18:01:00+01:00	60.0	1222684672.0
2017-12-22T18:02:00+01:00	60.0	1222394538.67
2017-12-22T18:03:00+01:00	60.0	1222147413.33
2017-12-22T18:01:20+01:00	1.0	1222684672.0
2017-12-22T18:01:30+01:00	1.0	1222684672.0
2017-12-22T18:01:40+01:00	1.0	1222684672.0
2017-12-22T18:01:50+01:00	1.0	1222684672.0
2017-12-22T18:02:00+01:00	1.0	1222684672.0
2017-12-22T18:02:10+01:00	1.0	1222684672.0
2017-12-22T18:02:20+01:00	1.0	1222684672.0
2017-12-22T18:02:30+01:00	1.0	1221943296.0
2017-12-22T18:02:40+01:00	1.0	1222438912.0
2017-12-22T18:02:50+01:00	1.0	1221931008.0
2017-12-22T18:03:00+01:00	1.0	1221931008.0
2017-12-22T18:03:10+01:00	1.0	1221931008.0
2017-12-22T18:03:20+01:00	1.0	1221931008.0
2017-12-22T18:03:30+01:00	1.0	1222373376.0
2017-12-22T18:03:40+01:00	1.0	1222369280.0
2017-12-22T18:03:50+01:00	1.0	1222348800.0

By default, metrics are stored with an archive policy set to “high”, which is defined to keep data as:

- Per second granularity for the last hour
- Per minute granularity for the last week
- Per hour granularity for a year

However, note that since `collectd` is configured to collect metrics only every 10 seconds, there is no metric measurement for each second but every 10 seconds.

16.2.1 Configuring `collectd`

While only a few `collectd` plugins are enabled by default, you can leverage the large collection of [available plugins](#). To enable a plugin on your instance, edit the instance's `/etc/collectd.conf` file. Uncomment each `LoadPlugin <plugin_name>` line that you wish to enable. Then, restart `collectd` with the command:

```
sudo systemctl restart collectd
```

The `collectd` configured to send measurements by batch to minimize network traffic. However, if you want to avoid any interference during your experiments, you can disable `collectd` with the command:

```
sudo systemctl stop collectd && sudo systemctl disable collectd
```

16.2.2 Metrics for bare metal nodes

Chameleon automatically collects power usage and temperature data on all nodes in the system. Instantaneous power usage data (in watts) and temperature readings (in Celsius) are collected through the IPMI interface on the chassis controller for the nodes. This “out-of-band” approach does not consume additional power on the node itself and runs even when the node is powered off.

Attention: Temperature metrics are currently collected from the CPU sensor on each node. These temperature readings are only reported while the node is powered on.

As with the system metrics, retrieving these automatically collected metrics for a node requires the OpenStack CLI and Gnocchi client plugin (see installation instructions [Setting up the Gnocchi CLI](#) above). To get a list of metrics available for a node, use this command:

```
$ gnocchi resource show <node_uuid>
```

To retrieve a specific reading:

```
$ gnocchi measures show <reading-name> --resource-id=<node_uuid> --refresh
```

Tip: The node UUID and the instance UUID are different. You can get a node's UUID for a reservation from the Horizon GUI (<https://chi.tacc.chameleoncloud.org> for TACC reservations, <https://chi.uc.chameleoncloud.org> for UC reservations). Click on your lease name from within the list of leases on the Leases subtab within the Reservations tab. The node UUID is at the very bottom under the Nodes section. You can also find an individual instance node UUID on the instance details page. Click on your instance name on the Instances tab and see `Physical Host Name`

For example, issuing the following command:

```
$ gnocchi measures show power --resource-id=05dd5e25-440f-4492-b3b8-9d39af83b8bc --
↪refresh
```

returns the following power results for node with id 05dd5e25-440f-4492-b3b8-9d39af83b8bc. The output below has been truncated:

timestamp	granularity	value
2018-03-21T07:00:00-05:00	3600.0	3.6990394736842047
2018-03-21T08:00:00-05:00	3600.0	3.6944069767441814
2018-03-21T09:00:00-05:00	3600.0	3.7072767295597435
2018-03-21T10:00:00-05:00	3600.0	3.6744999999999995
2018-03-21T11:00:00-05:00	3600.0	3.708236024844716
2018-03-21T12:00:00-05:00	3600.0	3.6747818181818137
2018-03-21T13:00:00-05:00	3600.0	3.706847058823526
.		
2018-05-07T08:17:43-05:00	1.0	3.537
2018-05-07T08:18:03-05:00	1.0	3.996
2018-05-07T08:18:23-05:00	1.0	3.847
2018-05-07T08:19:03-05:00	1.0	4.145
2018-05-07T08:19:23-05:00	1.0	4.145
2018-05-07T08:19:43-05:00	1.0	3.686
2018-05-07T08:20:03-05:00	1.0	3.847
2018-05-07T08:20:23-05:00	1.0	3.686
2018-05-07T08:20:43-05:00	1.0	3.847

To retrieve a metric for a specific time interval, pass the `start` and `stop` parameters; for example:

```
$ gnocchi measures show temperature_cpu --start 2018-11-27T02:00:00 --stop 2018-11-
↪27T03:00:00 --resource-id=f3f47a67-d805-48d4-9584-f0143ae976cf --refresh
```

returns:

timestamp	granularity	value
2018-11-27T02:00:00-06:00	300.0	61.0
2018-11-27T02:05:00-06:00	300.0	61.0
2018-11-27T02:10:00-06:00	300.0	61.0
2018-11-27T02:15:00-06:00	300.0	61.0
2018-11-27T02:20:00-06:00	300.0	58.6
2018-11-27T02:25:00-06:00	300.0	56.5333333333
2018-11-27T02:30:00-06:00	300.0	56.0
2018-11-27T02:35:00-06:00	300.0	56.0
2018-11-27T02:40:00-06:00	300.0	56.0
2018-11-27T02:45:00-06:00	300.0	56.0
2018-11-27T02:50:00-06:00	300.0	56.0
2018-11-27T02:55:00-06:00	300.0	56.0

16.2.3 Energy and Power Consumption Measurement with etrace2

The [CC-CentOS7](#), [CC-CentOS8](#), [CC-Ubuntu16.04](#) and [CC-Ubuntu18.04](#) appliances, as well as all Chameleon supported images derived from them, now include support for reporting energy and power consumption of each CPU socket and of memory DIMMs. It is done with the `etrace2` utility which relies on the [Intel RAPL \(Running Average Power Limit\)](#) interface.

Attention: Currently, `etrace2` requires a kernel feature that is not supported on our ARM nodes.

To spawn your program and print energy consumption:

```
etrace2 <your_program>
```

To print power consumption every 0.5 second:

```
etrace2 -i 0.5 <your_program>
```

To print power consumption every 1 second for 10 seconds:

```
etrace2 -i 1.0 -t 10
```

For example, to report energy consumption during the generation of a large RSA private key:

```
$ etrace2 openssl genrsa -out private.pem 4096
# ETRACE2_VERSION=0.1
Generating RSA private key, 4096 bit long modulus
.....
↪ .....
↪ .....
↪ .....++
.....
↪ .....++
e is 65537 (0x10001)
# ELAPSED=2.579472
# ENERGY=365.788208
# ENERGY_SOCKET0=99.037841
# ENERGY_DRAM0=78.577698
# ENERGY_SOCKET1=109.230103
# ENERGY_DRAM1=80.336548
```

The energy consumption is reported in joules.

`etrace2` reports power and energy consumption of CPUs and memory of the node during the entire execution of the program. This will include consumption of other programs running during this period, as well as power and energy consumption of CPUs and memory under idle load.

Note the following caveats:

- [Intel](#) documents that the RAPL is not an analog power meter, but rather uses a software power model. This software power model estimates energy usage by using hardware performance counters and I/O models. Based on their measurements, they match actual power measurements.
- In some situations the total `ENERGY` value is incorrectly reported as a value equal or close to zero. However, the sum of `ENERGY_SOCKET` and `ENERGY_DRAM` values should be accurate.

- Monitoring periods larger than 10-15 minutes may be inaccurate due to RAPL registers overflowing if they're not read regularly.

This [utility](#) was contributed by Chameleon user [Kazutomo Yoshii](#) of [Argonne National Laboratory](#).

Note: The Linux kernel version of [CC-Ubuntu16.04](#) is too old to use `etrace2` on Chameleon **Skylake** nodes. To solve the problem, simply upgrade the Linux kernel.

COMPLEX APPLIANCES

17.1 Introduction

Deploying an MPI cluster, an OpenStack installation, or any other type of cluster in which nodes can take on multiple roles can be complex: you have to provision potentially hundreds of nodes, configure them to take on various roles, and make them share information that is generated or assigned only at deployment time, such as hostnames, IP addresses, or security keys. When you want to run a different experiment later you have to redo all this work. When you want to reproduce the experiment, or allow somebody else to reproduce it, you have to take very precise notes and pay great attention to their execution.

To help solve this problem and facilitate reproducibility and sharing, the Chameleon team configured a tool that allows you to deploy complex clusters with “one click”. This tool requires not just a simple *image* (i.e., appliance) but also a document, called a *template*, that contains the information needed to orchestrate the deployment and configuration of such clusters. We call this *image + template* combination **Complex Appliances** because it consists of more than just the image (i.e., appliance).

In a nutshell, *Complex Appliances* allow you to specify not only what image you want to deploy but also on how many nodes you want to deploy that image, what roles the deployed instances should boot into (such as e.g., head node and worker node in a cluster), what information from a specific instance should be passed to another instance in that *Complex Appliance*, and what scripts should be executed on boot so that this information is properly used for configuring the “one click” cluster.

This guide will tell you all you need to know in order to use and configure *Complex Appliances* on Chameleon.

Hint: Since *Complex Appliances* in Chameleon are currently implemented using the [OpenStack Heat](#) orchestration service, we will be using OpenStack terminology and features to work with them. The templates described above are YAML files using the [Heat Orchestration Template \(HOT\)](#) format (Heat also supports the AWS CloudFormation template format, but this is not covered here). A deployed complex appliance is referred to as a “stack” – just as a deployed single appliance is typically referred to as an “instance”.

17.2 Complex Appliances in the Catalog

The [Chameleon Appliance Catalog](#) has several *Complex Appliances* for popular technologies that people want to deploy such as OpenStack or MPI or even more advanced deployments such as efficient SR-IOV enabled MPI in KVM virtual machines. We also provide common building blocks for cluster architectures, such as an NFS share. *Complex Appliances* are identified by a badge in their top-right corner representing a group of machines, as in the screenshot below:

To view the details of a *Complex Appliance*, simply click on it.

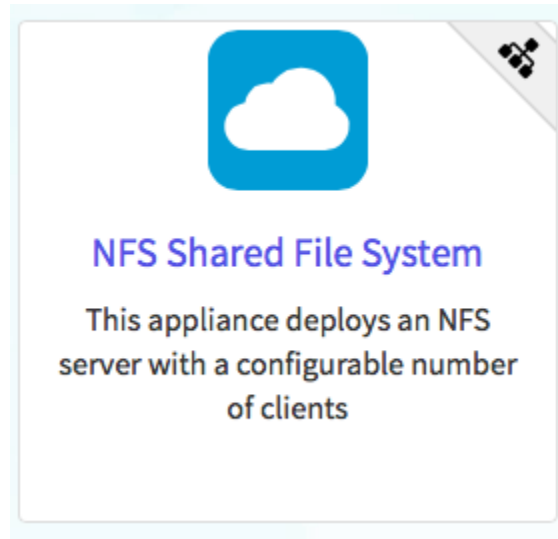


Fig. 1: A Complex Appliance with a badge in the upper right



NFS share

[Launch Complex Appliance at CHI@UC](#)
[Launch Complex Appliance at CHI@TACC](#)
[Edit](#)

Description

This appliance deploys an NFS server with a configurable number of clients. The NFS server creates a directory at `/exports/example`, changes its ownership to the "cc" user and group, and exports it. The NFS clients add this NFS share to `/etc/fstab` and mount it.

This appliance accepts the following parameters:

- `nfs_client_count`: Number of NFS client instances (defaults to 1)
- `key_name`: name of a key pair to enable SSH access to the instance (defaults to "default")
- `reservation_id`: ID of the Blazar reservation to use for launching instances

The following outputs are provided:

- `server_ip`: the public IP address of the NFS server
- `client_ips`: the private IP addresses of the NFS clients


[Chameleon Supported](#)

Template

[Get Template](#)

Author

Name: Chameleon Project
Contact: help@chameleoncloud.org

Fig. 2: A Complex Appliance page

Tip: You may download the *Template* file or copy the *Template* file URL to clipboard by clicking the *Get Template* button. The *Template* file or it's URL is required when launching a *Complex Appliance*.

17.3 Managing Complex Appliances using the GUI

Before launching a *Complex Appliance*, please make sure that you have a reservation for the appropriate node types and a key pair configured. Since most *Complex Appliances* will consist of multiple nodes, make sure you have set the *Minimum Number of Hosts* in your Lease. You will also need a *Template* file or the URL for a *Template* file from the [Appliance Catalog](#). At CHI@TACC site or CHI@UC site, go to *Project > Orchestration > Stacks* use the navigation side bar.

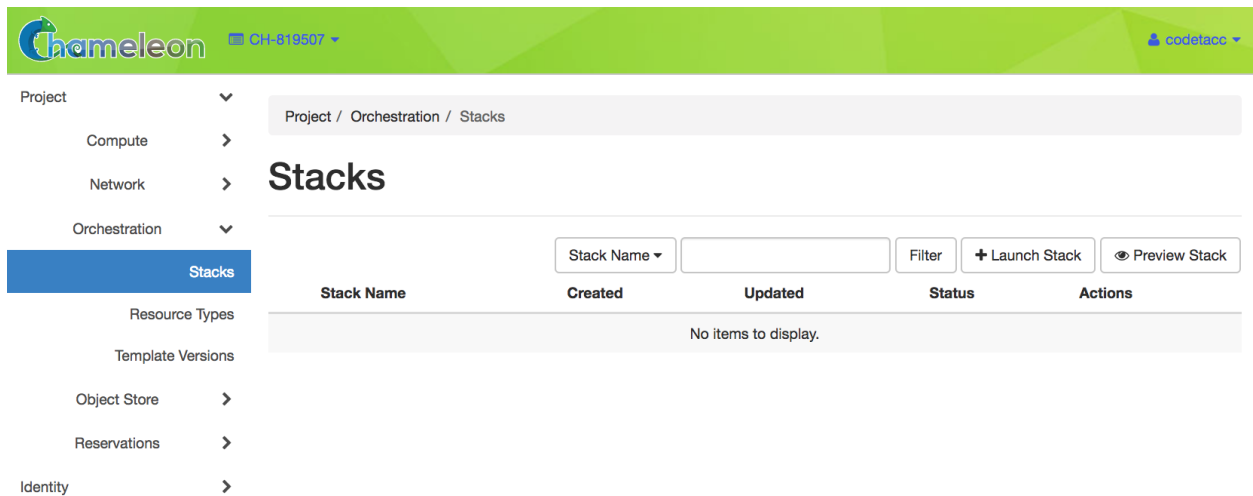


Fig. 3: The Stacks page

Tip: You can go to *Stacks* page directly from the [Appliance Catalog](#).

1. Go to the [Appliance Catalog](#) and identify the appliance you want to launch. Click on it to open its details page.
2. Click on the “Launch Complex Appliance at CHI\@TACC” or “Launch Complex Appliance at CHI\@UC” button depending on where your reservation is created.

17.3.1 Launching a Complex Appliance

To launch a stack, click the *Launch Stack* button in the upper right of the *Stacks* page. Then follow the steps:

1. Start setting up a *Template* by choosing a *Template Source* in the dropdown. You may either select the *File* option as *Template Source* and upload the *Template* file, or select the *URL* option and provide the URL of the *Template* file.

Important: Do not change the environment source settings!

Fig. 4: The Select Template step

2. Once you have provided a Template, click the *Next* button. Chameleon will validate the Template file and proceed to the *Launch Stack* step.
3. Choose a name for your stack. Ignore the “Creation Timeout” and “Rollback On Failure” settings. You also need to enter your Chameleon password. Then, you need to select a value for the parameters of the template. Finally, click the *Launch* button.
4. Your stack should be in status “Create In Progress” for several minutes while it first launches the server instance, followed by the client instances. It will then move to the status “Create Complete”.

17.3.2 Monitoring a Complex Appliance

To monitor and get more details about your *Complex Appliance*, click on it in the *Stacks* page.

- The *Topology* tab displays a topology graph of the stack. The rack of machine represents the client instance group. The server’s floating IP (the public IP assigned to a resource) is represented by an IP in a circle; while an IP in a circle is also used to represent the association of the IP with the server instance (not the greatest idea to use the same symbol for both the IP and the association – we agree but can’t do much about it at the moment). Blow off some steam by dragging the visualization across the screen, it can be rather fun!

Note: Blinking nodes indicates that they are still provisioning.

- The *Overview* tab displays various parameters, including the *ID* of the stack and *Outputs* such as IP addresses assigned to each node. If you have a floating IP associated to the server, you can now `ssh` to the server using the floating IP just as you do with regular instances. The client may not have a floating IP attached to it, but you can connect to it via the server node with the client’s private IP.

Launch Stack

Stack Name * ?

Creation Timeout (minutes) * ?

60

☐ Rollback On Failure ?

Password for user "codetacc" * ?

key_name ?

Select a key pair

nfs_client_count ?

1

reservation_id * ?

Select Reservation

Description:

Create a new stack with the provided values.

Cancel

Launch

Fig. 5: The Launch Stack step

Chameleon

CH-819507

codetacc

Project

Compute

Network

Orchestration

Stacks

Resource Types

Template Versions

Object Store

Reservations

Identity

Project / Orchestration / Stacks

Stacks

Stack Name

Filter

+ Launch Stack

Preview Stack

Delete Stacks

More Actions

Displaying 1 item

<input type="checkbox"/> Stack Name	Created	Updated	Status	Actions
<input type="checkbox"/> complex_appliance	0 minutes	Never	Create In Progress	Check Stack

Displaying 1 item

Fig. 6: A Complex Appliance with the Create in Progress status

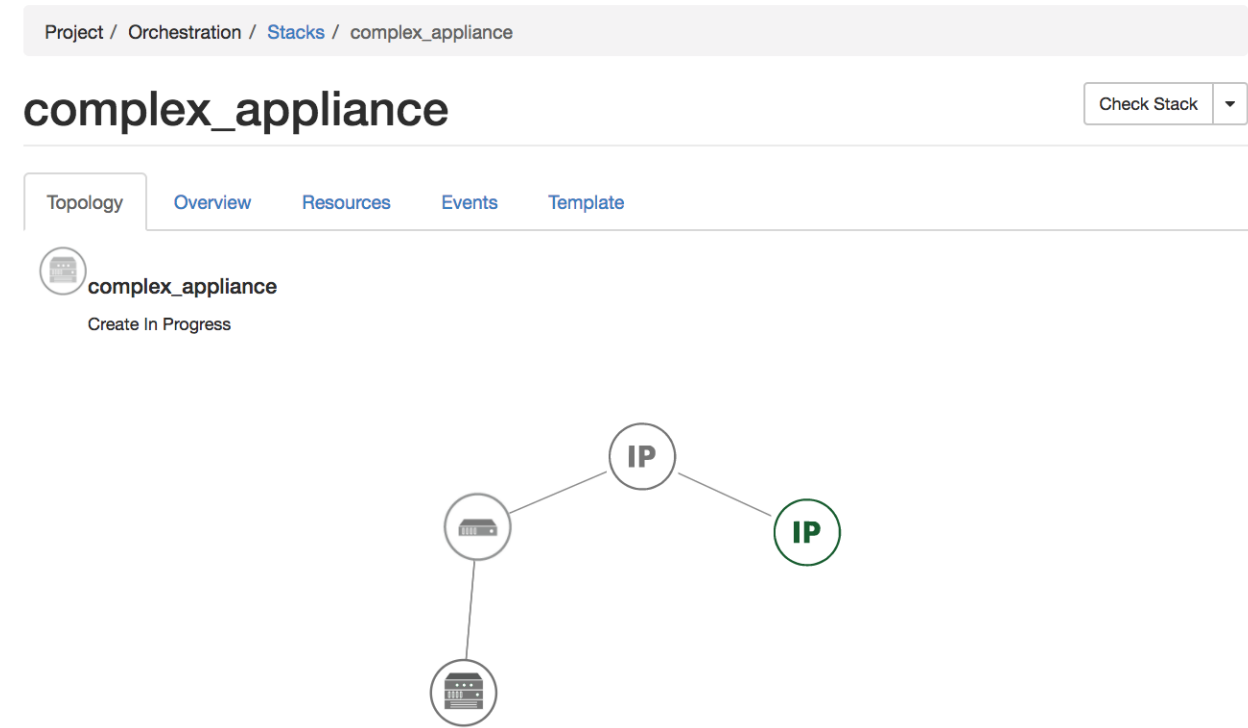


Fig. 7: The Topology tab

Tip: To talk to the client without an associated floating IP, connect to the server with `ssh -A` to enable the SSH agent forwarding after loading your key to your SSH agent with `ssh-add <path-to-your-key>`.

- Under the *Resources* tab you will see the resources of the stack (the server, clients, server's public/floating IP, and its the association) and information about them.
- In the *Events* tab you will see information about the history of the deployment so far.
- In *Template* tab, you will see the template that was used to deploy this stack.

17.3.3 Deleting a Complex Appliance

To delete a *Complex Appliance*, select it in the *Stacks* page and click the *Delete Stacks* button. This will delete all resources of the stack, such as nodes and floating IP addresses.

complex_appliance

[Check Stack](#)
[Topology](#)
[Overview](#)
[Resources](#)
[Events](#)
[Template](#)

Name complex_appliance
ID a74457ce-51f2-41b4-9ccf-cb10fbbcca08
Description NFS server and clients deployed with Heat on Chameleon

Status

Created 13 minutes
Last Updated Never
Status Create_Complete: Stack CREATE completed successfully

Outputs

client_ips Private IP addresses of the NFS clients

```
[
  "10.52.1.72",
  "10.52.1.39"
]
```

server_ip Public IP address of the NFS server

```
129.114.108.167
```

Fig. 8: The Overview tab

Project / Orchestration / [Stacks](#) / complex_appliance

complex_appliance

[Check Stack](#)
[Topology](#)
[Overview](#)
[Resources](#)
[Events](#)
[Template](#)

Displaying 4 items

Stack Resource	Resource	Stack Resource Type	Date Updated	Status	Status Reason
nfs_server_ip_association		OS::Nova::FloatingIPAssociation	5 minutes	Init Complete	
nfs_server	74558264-f1a8-45f3-8a88-38b9c98984ea	OS::Nova::Server	5 minutes	Create In Progress	state changed
nfs_server_floating_ip	61fcb3b9-93a9-4bea-a03e-81bff0c81613	OS::Nova::FloatingIP	5 minutes	Create Complete	state changed
nfs_clients		OS::Heat::ResourceGroup	5 minutes	Init Complete	

Displaying 4 items

Fig. 9: The Resources tab

Project / Orchestration / [Stacks](#) / complex_appliance

complex_appliance

Check Stack ▾

Topology

Overview

Resources

Events

Template

Displaying 8 items

Stack Resource	Resource	Time Since Event	Status	Status Reason
nfs_server_ip_association	13943	0 minutes	Create Complete	state changed
nfs_clients	complex_appliance-nfs_clients-lpwemi2expf5	0 minutes	Create In Progress	state changed
nfs_server_ip_association	complex_appliance-nfs_server_ip_association-2cboxikqitcu	0 minutes	Create In Progress	state changed
nfs_server	74558264-f1a8-45f3-8a88-38b9c98984ea	0 minutes	Create Complete	state changed
nfs_server_floating_ip	61fcb3b9-93a9-4bea-a03e-81bff0c81613	6 minutes	Create Complete	state changed

Fig. 10: The Events tab

Project / Orchestration / [Stacks](#) / complex_appliance

complex_appliance

Check Stack ▾

Topology

Overview

Resources

Events

Template

```
description: NFS server and clients deployed with Heat on Chameleon
heat_template_version: '2015-10-15'
outputs:
  client_ips:
    description: Private IP addresses of the NFS clients
    value:
      get_attr: [nfs_clients, first_address]
  server_ip:
    description: Public IP address of the NFS server
    value:
      get_attr: [nfs_server_floating_ip, ip]
parameters:
  key_name:
    constraints:
      - {custom_constraint: nova.keypair}
    default: default
    description: Name of a KeyPair to enable SSH access to the instance
    type: string
  nfs_client_count:
    constraints:
```

Fig. 11: The Template tab

17.4 Managing Complex Appliances using the CLI

Tip: Reading *Command Line Interface (CLI)* is highly recommended before continuing on the following sections.

In addition to *Installing the CLI*, you will need to install the `python-heatclient` package using the command:

```
pip install python-heatclient
```

Then, set up your environment for OpenStack command line usage, as described in *The OpenStack RC Script*. You can get a list of your *Complex Appliances* in your project using the command:

```
openstack stack list
```

The output should look like the following:

ID	Stack Name	Stack Status	Creation Time
Updated Time			
e5df33b5-5282-4935-8097-973328ca71e5	my_stack	CREATE_COMPLETE	2018-01-23T22:45:12Z
None			

17.4.1 Launching a Complex Appliance

To launch a *Complex Appliance* using *Template*, run the command on your local machine:

```
openstack stack create --template <template_file> --parameter <parameter>=<value> <stack_name>
```

Provide the path to and the name of the *Template* file in your local file system via the `template` switch. The `<stack_name>` is the name of the *Complex Appliance*. In addition, you may provide the parameters required in the *Template* file with their values by parameter switch. For example, the *NFS Server Template* lists the following parameters section:

```
parameters:
  nfs_client_count:
    type: number
    description: Number of NFS client instances
    default: 1
    constraints:
      - range: { min: 1 }
      description: There must be at least one client.
  key_name:
    type: string
    description: Name of a KeyPair to enable SSH access to the instance
    default: default
    constraints:
```

(continues on next page)

(continued from previous page)

```
- custom_constraint: nova.keypair
reservation_id:
  type: string
  description: ID of the Blazar reservation to use for launching instances.
  constraints:
    - custom_constraint: blazar.reservation
```

Therefore, in order to use this *Template*, you must provide values for `nfs_client_count`, `key_name` and `reservation_id`.

17.4.2 Monitoring a Complex Appliance

You can get details about your *Complex Appliance*, such as *Outputs*, *Events* and *Resources*, via the CLI. You will need the *UUID* of the *Complex Appliance*.

Tip: To get the *UUID* of your *Complex Appliance*, use the *Stacks* page on the GUI or retrieve it by `openstack stack list` command.

- To view the *Outputs*, run:

```
openstack stack output list <uuid>
```

For example, the list of the outputs for the *NFS Share* stack is:

```
+-----+-----+
| output_key | description |
+-----+-----+
| client_ips | Private IP addresses of the NFS clients |
| server_ip  | Public IP address of the NFS server   |
+-----+-----+
```

You can get more details about the outputs by using the following command:

```
openstack stack output show --all <uuid>
```

- To view the *Events*, run:

```
openstack stack event list <uuid>
```

- To view the *Resources*, run:

```
openstack stack resource list <uuid>
```

Your output may look like this:

```
+-----+-----+-----+-----+
↪ | resource_name | physical_resource_id | resource_type ↪
↪ |               | resource_status    | updated_time   |
+-----+-----+-----+-----+
↪ +-----+-----+-----+-----+
```

(continues on next page)

(continued from previous page)

```

| nfs_server_ip_association |                               |
↪ OS::Neutron::FloatingIPAssociation | INIT_COMPLETE   | 2018-03-19T18:38:05Z |
| nfs_server                | 0ab0169c-f762-4d27-8724-b359caa50f1f |
↪ OS::Nova::Server          | CREATE_FAILED   | 2018-03-19T18:38:05Z |
| nfs_server_floating_ip    | ecb391f8-4653-43a6-b2c6-bb93a6d89115 |
↪ OS::Nova::FloatingIP      | CREATE_COMPLETE | 2018-03-19T18:38:05Z |
| nfs_clients               |                               |
↪ OS::Heat::ResourceGroup   | INIT_COMPLETE   | 2018-03-19T18:38:05Z |
+-----+-----+-----+-----+-----+
↪ -----+-----+-----+-----+-----+

```

Then, you may retrieve information about a specific resource using the command:

```
openstack stack resource show <stack_uuid> <resource_name>
```

17.4.3 Deleting a Complex Appliance

Use the following command to delete a stack:

```
openstack stack delete <uuid>
```

It will remove all the resources attached to the stack.

17.5 Heat Orchestration Templates

A *Heat Orchestration Template* is a YAML file that specifies how resources are used and configured in a *Complex Appliance*.

17.5.1 A Case Example: NFS Share

Let's look at the [NFS Share Template](#). The NFS share appliance deploys:

- An NFS server instance, that exports the directory `/exports/example` to any instance running on Chameleon bare metal,
- One or several NFS client instances, which configure `/etc/fstab` to mount this NFS share to `/mnt` (and can subsequently read from and write to it).

This template is reproduced further below, and includes inline comments starting with the `#` character. There are three main sections:

- resources
- parameters
- outputs

The `resources` section is the most important part of the template: it defines which OpenStack *Resources* to create and configure. Inside this section you can see four resources defined:

- `nfs_server_floating_ip`: creates a *Floating IP* on the `ext-net` public network. It is not attached to any instance yet.

- `nfs_server`: creates the NFS server instance (an instance is defined with the type `OS::Nova::Server` in *Heat*). It is a bare metal instance (`flavor: baremetal`) using the CC-CentOS7 image and connected to the private network named `sharednet1`. We set the key pair to use the value of the parameter defined earlier, using the `get_param` function. Similarly, the reservation to use is passed to the scheduler. Finally, a `user_data` script is given to the instance, which configures it as an NFS server exporting `/exports/example` to Chameleon instances.
- `nfs_server_ip_association`: associates the floating IP created earlier with the NFS server instance.
- `nfs_clients`: defines a resource group containing instance configured to be NFS clients and mount the directory exported by the NFS server defined earlier. The IP of the NFS server is gathered using the `get_attr` function, and placed into `user_data` using the `str_replace` function.

Once a Resource has been specified, you may provide it as a value for another Resource's property using the `get_resource` function.

The `parameters` section defines inputs to be used on *Complex Appliance* launch. Parameters all have the same data structure: each one has a name (`key_name` or `reservation_id` in this case), a data type (`number` or `string`), a comment field called `description`, optionally a default value, and a list of constraints (in this case only one per parameter). Constraints tell *Heat* to match a parameter to a specific type of OpenStack resource. *Complex appliances* on Chameleon require users to customize at least the key pair name and reservation ID, and will generally provide additional parameters to customize other properties of the cluster, such as its size, as in this example. The values of Parameters can be used in the `resources` section using the `get_param` function.

The `outputs` section defines what values are returned to the user. *Outputs* are declared similarly to *Parameters*: they each have a name, an optional description, and a value. They allow to return information from the stack to the user. You may use the `get_attr` function to retrieve a resource's attribute for output.

17.5.2 Heat Template Customization

Customizing an existing template is a good way to start developing your own. We will use a simpler template than the previous example to start with: it is the [Hello World complex appliance](#).

First, delete the stack you launched, because we will need all three nodes to be free. To do this, go back to the *Project > Orchestration > Stacks* page, select your stack, and then click on the *Delete Stacks* button. You will be asked to confirm, so click on the *Delete Stacks* button.

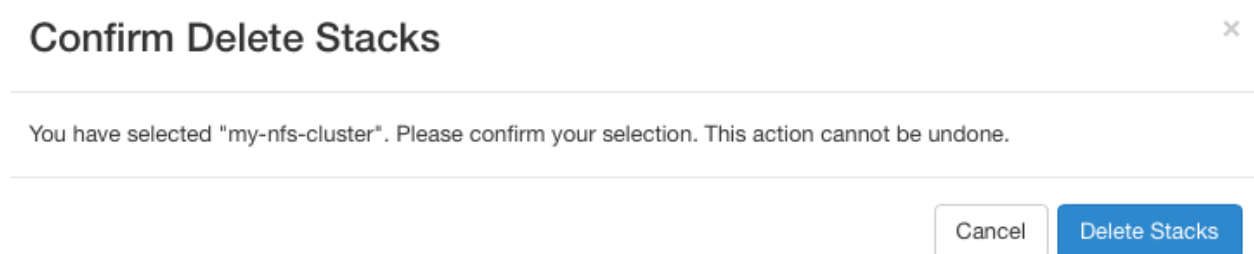


Fig. 12: Confirm deleting stack dialog

The template for the [Hello World complex appliance](#) is reproduced below. It is similar to the NFS share appliance, except that it deploys only a single client. You can see that it has four resources defined:

- `nfs_server_floating_ip`
- `nfs_server`

- nfs_server_ip_association
- nfs_client

The `nfs_client` instance mounts the NFS directory shared by the `nfs_server` instance, just like in our earlier example.

```
# This describes what is deployed by this template.
description: NFS server and client deployed with Heat on Chameleon

# This defines the minimum Heat version required by this template.
heat_template_version: 2015-10-15

# The resources section defines what OpenStack resources are to be deployed and
# how they should be configured.
resources:
  nfs_server_floating_ip:
    type: OS::Nova::FloatingIP
    properties:
      pool: ext-net

  nfs_server:
    type: OS::Nova::Server
    properties:
      flavor: baremetal
      image: CC-CentOS7
      key_name: { get_param: key_name }
      networks:
        - network: sharednet1
      scheduler_hints: { reservation: { get_param: reservation_id } }
      user_data: |
        #!/bin/bash
        yum install -y nfs-utils
        mkdir -p /exports/example
        chown -R cc:cc /exports
        echo '/exports/example 10.140.80.0/22(rw,async) 10.40.0.0/23(rw,async)' >> /etc/
↪ exports
        systemctl enable rpcbind && systemctl start rpcbind
        systemctl enable nfs-server && systemctl start nfs-server

  nfs_server_ip_association:
    type: OS::Neutron::FloatingIPAssociation
    properties:
      floatingip_id: {get_resource: nfs_server_floating_ip}
      port_id: {get_attr: [nfs_server, addresses, sharednet1, 0, port]}

  nfs_client:
    type: OS::Nova::Server
    properties:
      flavor: baremetal
      image: CC-CentOS7
      key_name: { get_param: key_name }
      networks:
        - network: sharednet1
```

(continues on next page)

(continued from previous page)

```

scheduler_hints: { reservation: { get_param: reservation_id } }
user_data:
  str_replace:
    template: |
      #!/bin/bash
      yum install -y nfs-utils
      echo "$nfs_server_ip:/exports/example    /mnt/    nfs" > /etc/fstab
      mount -a
    params:
      $nfs_server_ip: { get_attr: [nfs_server, first_address] }

# The parameters section gathers configuration from the user.
parameters:
  key_name:
    type: string
    description: Name of a KeyPair to enable SSH access to the instance
    default: default
    constraints:
      - custom_constraint: nova.keypair
  reservation_id:
    type: string
    description: ID of the Blazar reservation to use for launching instances.
    constraints:
      - custom_constraint: blazar.reservation

```

Download [this template](#) to your local machine, and open it in your favorite text editor.

We will customize the template to add a second NFS client by creating a new resource called `another_nfs_client`. Add the following text to your template inside the resources section. Make sure to respect the level of indentation, which is important in YAML.

```

another_nfs_client:
  type: OS::Nova::Server
  properties:
    flavor: baremetal
    image: CC-CentOS7
    key_name: { get_param: key_name }
    networks:
      - network: sharednet1
  scheduler_hints: { reservation: { get_param: reservation_id } }
  user_data:
    str_replace:
      template: |
        #!/bin/bash
        yum install -y nfs-utils
        echo "$nfs_server_ip:/exports/example    /mnt/    nfs" > /etc/fstab
        mount -a
      params:
        $nfs_server_ip: { get_attr: [nfs_server, first_address] }

```

Now, launch a new stack with this template. Since the customized template is only on your computer and cannot be addressed by a URL, use the *Direct Input* method instead and copy/paste the content of the customized template. The resulting topology view is shown below: as you can see, the two client instances are shown separately since each one

is defined as a separate resource in the template.

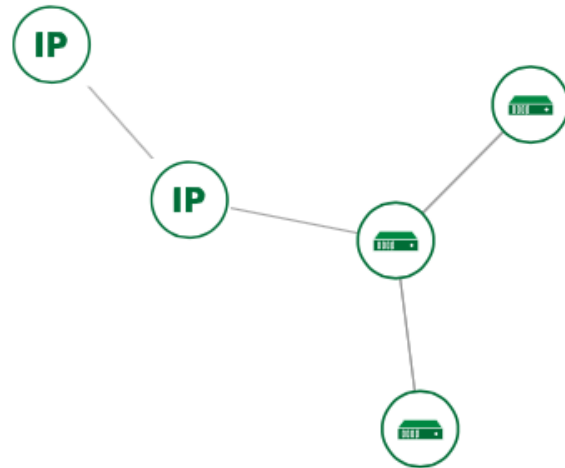
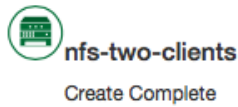


Fig. 13: Topology of the customized Hello World Appliance

You may have realized already that while adding just one additional client instance was easy, launching more of them would require to copy / paste blocks of YAML many times while ensuring that the total count is correct. This would be easy to get wrong, especially when dealing with tens or hundreds of instances.

So instead, we leverage another construct from *Heat*: resource groups. Resource groups allow to define one kind of resource and request it to be created any number of times.

Remove the `nfs_client` and `another_client` resources from your customized template, and replace them with the following:

```
nfs_clients:
  type: OS::Heat::ResourceGroup
  properties:
    count: 2
    resource_def:
      type: OS::Nova::Server
      properties:
        flavor: baremetal
        image: CC-CentOS7
        key_name: { get_param: key_name }
        networks:
          - network: sharednet1
        scheduler_hints: { reservation: { get_param: reservation_id } }
        user_data:
```

(continues on next page)

(continued from previous page)

```

str_replace:
  template: |
    #!/bin/bash
    yum install -y nfs-utils
    echo "$nfs_server_ip:/exports/example    /mnt/    nfs" > /etc/fstab
    mount -a
  params:
    $nfs_server_ip: { get_attr: [nfs_server, first_address] }

```

A resource group is configured with a `properties` field, containing the definition of the resource to launch (`resource_def`) and the number of resources to launch (`count`). Once launched, you will notice that the topology view groups all client instances under a single *Resource Group* icon. We use the same `resource_def` than when defining separate instances earlier.

Another way we can customize this template is by adding outputs to the template. Outputs allow a *Heat* template to return data to the user. This can be useful to return values like IP addresses or credentials that the user must know to use the system.

We will create an output returning the floating IP address used by the NFS server. We define an `outputs` section, and one output with the name `server_ip` and a description. The value of the output is gathered using the `get_attr` function which obtains the IP address of the server instance.

```

outputs:
  server_ip:
    description: Public IP address of the NFS server
    value: { get_attr: [nfs_server_floating_ip, ip] }

```

You can get outputs in the *Overview* tab of the *Stack Details* page. If you want to use the command line, install `python-heatclient` and use the `heat output-list` and `heat output-show` commands, or get a full list in the information returned by `heat stack-show`.

Multiple outputs can be defined in the `outputs` section. Each of them needs to have a unique name. For example, we can add another output to list the private IPs assigned to client instances:

```

client_ips:
  description: Private IP addresses of the NFS clients
  value: { get_attr: [nfs_clients, first_address] }

```

The image below shows the resulting outputs as viewed from the web interface. Of course IP addresses will be specific to each deployment.

Finally, we can add a new parameter to replace the hard-coded number of client instances by a value passed to the template. Add the following text to the `parameters` section:

```

nfs_client_count:
  type: number
  description: Number of NFS client instances
  default: 1
  constraints:
    - range: { min: 1 }
      description: There must be at least one client.

```

Inside the resource group definition, change `count: 2` to `count: { get_param: nfs_client_count }` to retrieve and use the parameter we just defined. When you launch this template, you will see that an additional parameter

Outputs	
client_ips	Private IP address of the NFS clients
	<pre>["10.140.82.20", "10.140.82.19"]</pre>
server_ip	Public IP address of the NFS server
	<pre>130.202.88.157</pre>

Fig. 14: The Outputs of customized Hello World appliance

allows you to define the number of client instances, like in the NFS share appliance.

At this stage, we have fully recreated the *NFS share* appliance starting from the *Hello World* one! The next section will explain how to write a new template from scratch.

17.5.3 Writing a New Template

You may want to write a whole new template, rather than customizing an existing one. Each template should follow the same layout and be composed of the following sections:

- Heat template version
- Description
- Resources
- Parameters
- Outputs

Heat template version

Each Heat template has to include the `heat_template_version` key with a valid version of [HOT \(Heat Orchestration Template\)](#). Chameleon bare metal supports any HOT version up to **2015-10-15**, which corresponds to OpenStack Liberty. The [Heat documentation](#) lists all available versions and their features. We recommended that you always use the latest Chameleon supported version to have access to all supported features:

```
heat_template_version: 2015-10-15
```

Description

While not mandatory, it is good practice to describe what is deployed and configured by your template. It can be on a single line:

```
description: This describes what this Heat template deploys on Chameleon.
```

If a longer description is needed, you can provide multi-line text in YAML, for example:

```
description: >
  This describes what this Heat
  template deploys on Chameleon.
```

Resources

The resources section is required and must contain at least one resource definition. A [complete list of resources types known to Heat](#) is available.

However, only a subset of them are supported by Chameleon, and some are limited to administrative use. We recommend that you only use:

- OS::Glance::Image
- OS::Heat::ResourceGroup
- OS::Heat::SoftwareConfig
- OS::Heat::SoftwareDeployment
- OS::Heat::SoftwareDeploymentGroup
- OS::Neutron::FloatingIP
- OS::Neutron::FloatingIPAssociation
- OS::Neutron::Port (advanced users only)
- OS::Nova::Keypair
- OS::Nova::Server

If you know of another resource that you would like to use and think it should be supported by the OpenStack services on Chameleon bare metal, please let us know via our [Help Desk](#).

Parameters

Parameters allow users to customize the template with necessary or optional values. For example, they can customize which Chameleon appliance they want to deploy, or which key pair to install. Default values can be provided with the `default` key, as well as constraints to ensure that only valid OpenStack resources can be selected. For example, `custom_constraint: glance.image` restricts the image selection to an available OpenStack image, while providing a pre-filled selection box in the web interface. [More details about constraints](#) are available in the *Heat* documentation.

Outputs

Outputs allow template to give information from the deployment to users. This can include usernames, passwords, IP addresses, hostnames, paths, etc. The outputs declaration is using the following format:

```
outputs:
  first_output_name:
    description: Description of the first output
    value: first_output_value
  second_output_name:
    description: Description of the second output
    value: second_output_value
```

Generally values will be calls to `get_attr`, `get_param`, or some other function to get information from parameters or resources deployed by the template and return them in the proper format to the user.

17.5.4 Reserved Networks and Floating IPs

Chameleon's reservation service allows users to reserve VLAN segments and floating ips. In order to make use of these reserved resources in a (HOT) template, follow the guidelines below. For more information on VLAN and floating ip reservations, see documentaiton on [Creating a Lease to Reserve a VLAN Segment](#) and [Creating a Lease to Reserve Floating IPs](#)

When you reserve a VLAN segment via blazar, it will automatically create a network for you. However, this network is not usable in your template unless a subnet and router have been associated with the network. Once this is done, you can simply add the network name as the network parameter for your server as you would `sharednet1`. The below cli commands provides an example of how to complete the setup for your reserved network.

```
openstack subnet create --subnet-range 192.168.100.0/24 \
  --allocation-pool start=192.168.100.100,end=192.168.100.108 \
  --dns-nameserver 8.8.8.8 --dhcp \
  --network <my_reserved_network_name> \
  my_subnet_name
openstack router create my_router_name
openstack router add subnet my_router_name my_subnet_name
openstack router set --external-gateway public my_router_name
```

For reserved floating ips, you need to associate the floating ip with a server using the `OS::Neutron::FloatingIPAssociation` object type. Many of our older complex appliance templates use the `OS::Nova::FloatingIPAssociation` object, but this has since been deprecated. See example below for proper usage:

```
my_server_ip_association:
  type: OS::Neutron::FloatingIPAssociation
  properties:
    floatingip_id: <my_reserved_floating_ip_uuid>
    port_id: {get_attr: [my_server, addresses, <my_network_name>, 0, port]}
```

If you are having trouble finding the uuid of the floating ip address then the below command will help you.

```
openstack floating ip list -c ID -c "Floating IP Address" -c Tags --long
```

The output should look like the sample output below with the *uuid* listed under the *ID* column. You can check your lease in the reservation section of the GUI to find the *reservation id* associated with the floating ip in the *Tags* section of the output.

ID	Floating IP Address	Tags
0fe31fad-60ac-462f-bb6c-4d40c1506621	192.5.87.206	[u'reservation:d90ad917-300a-4cf7-a836-083534244f56', u'blazar']
92a347a9-31a5-43c1-80e2-9cdb38ebf66f	192.5.87.224	[u'reservation:5f470c97-0166-4934-a813-509b743e2d62', u'blazar']
c8480d67-533d-4f55-a197-8271da6d9344	192.5.87.71	[]

17.6 Sharing Complex Appliances

If you have written your own *Complex Appliance* or substantially customized an existing one, we would love if you shared them with our user community! The process is very similar to regular appliances: log into the Chameleon portal, go to the appliance catalog, and click on the button in the top-right corner: *Add an appliance* (you need to be logged in to see it).

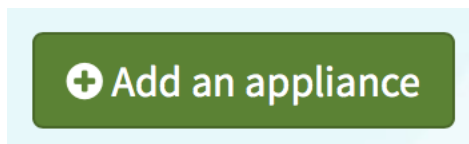


Fig. 15: The Add an Appliance button

You will be prompted to enter a name, description, and documentation. Instead of providing appliance IDs, copy your template to the dedicated field. Finally, share your contact information and assign a version string to your appliance. Once submitted, your appliance will be reviewed. We will get in touch if a change is needed, but if it's all good we will publish it right away!

17.7 Advanced Topics

17.7.1 All-to-All Information Exchange

The previous examples have all used `user_data` scripts to provide instances with contextualization information. While it is easy to use, this contextualization method has a major drawback: because it is given to the instance as part of its launch request, it cannot use any context information that is not yet known at this time. In practice, this means that in a client-server deployment, only one of these pattern will be possible:

- The server has to be deployed first, and once it is deployed, the clients can be launched and contextualized with information from the server. The server won't know about the clients unless there is a mechanism (not managed by *Heat*) for the client to contact the server.
- The clients have to be deployed first, and once they are deployed, the server can be launched and contextualized with information from the clients. The clients won't know about the server unless there is a mechanism (not managed by *Heat*) for the server to contact the clients.

This limitation was already apparent in our [NFS share](#) appliance: this is why the server instance exports the file system to all bare metal instances on Chameleon, because it doesn't know which specific IP addresses are allocated to the clients.

This limitation is even more important if the deployment is not hierarchical, i.e. all instances need to know about all others. For example, a cluster with IP and hostnames populated in `/etc/hosts` required each instance to be known by every other instance.

This section presents a more advanced form of contextualization that can perform this kind of information exchange. This is implemented by *Heat* agents running inside instances and communicating with the *Heat* service to send and receive information. This means you will need to use an image bundling these agents. Currently, all Chameleon-supported images (CC) are supporting this mode of contextualization. If you build your own images using the [CC-CentOS7](#) builder, [CC-CentOS](#) builder or [CC-Ubuntu](#) builder, you will automatically have these agents installed. This contextualization is performed with several Heat resources:

- `OS::Heat::SoftwareConfig`: This resource describes code to run on an instance. It can be configured with inputs and provide outputs.
- `OS::Heat::SoftwareDeployment`: This resource applies a `SoftwareConfig` to a specific instance.
- `OS::Heat::SoftwareDeploymentGroup`: This resource applies a `SoftwareConfig` to a specific group of instances.

The template below illustrates how it works. It launches a group of instances that will automatically populate their `/etc/hosts` file with IP and hostnames from other instances in the deployment.

```
heat_template_version: 2015-10-15

description: >
  This template demonstrates how to exchange hostnames and IP addresses to populate /etc/
  ↪hosts.

parameters:
  flavor:
    type: string
    default: baremetal
    constraints:
      - custom_constraint: nova.flavor
  image:
    type: string
    default: CC-CentOS8
    constraints:
      - custom_constraint: glance.image
  key_name:
    type: string
    default: default
    constraints:
      - custom_constraint: nova.keypair
  instance_count:
    type: number
    default: 2
  reservation_id:
    type: string
    description: ID of the Blazar reservation to use for launching instances.
    constraints:
      - custom_constraint: blazar.reservation
```

(continues on next page)

(continued from previous page)

```

resources:
  export_hosts:
    type: OS::Heat::SoftwareConfig
    properties:
      outputs:
        - name: hosts
      group: script
      config: |
        #!/bin/sh
        (echo -n $(factor ipaddress); echo -n ' '; echo $(factor hostname)) > ${heat_
↪outputs_path}.hosts

  export_hosts_sdg:
    type: OS::Heat::SoftwareDeploymentGroup
    properties:
      config: { get_resource: export_hosts }
      servers: { get_attr: [server_group, refs_map] }
      signal_transport: HEAT_SIGNAL

  populate_hosts:
    type: OS::Heat::SoftwareConfig
    properties:
      inputs:
        - name: hosts
      group: script
      config: |
        #!/usr/bin/env python
        import ast
        import os
        import string
        import subprocess
        hosts = os.getenv('hosts')
        if hosts is not None:
            hosts = ast.literal_eval(string.replace(hosts, '\n', '\\n'))
        with open('/etc/hosts', 'a') as hosts_file:
            for ip_host in hosts.values():
                hosts_file.write(ip_host.rstrip() + '\n')

  populate_hosts_sdg:
    type: OS::Heat::SoftwareDeploymentGroup
    depends_on: export_hosts_sdg
    properties:
      config: { get_resource: populate_hosts }
      servers: { get_attr: [server_group, refs_map] }
      signal_transport: HEAT_SIGNAL
      input_values:
        hosts: { get_attr: [ export_hosts_sdg, hosts ] }

  server_group:
    type: OS::Heat::ResourceGroup
    properties:

```

(continues on next page)

(continued from previous page)

```

count: { get_param: instance_count }
resource_def:
  type: OS::Nova::Server
  properties:
    flavor: { get_param: flavor }
    image: { get_param: image }
    key_name: { get_param: key_name }
    networks:
      - network: sharednet1
    scheduler_hints: { reservation: { get_param: reservation_id } }
    user_data_format: SOFTWARE_CONFIG
    software_config_transport: POLL_SERVER_HEAT

outputs:
  deployment_results:
    value: { get_attr: [export_hosts_sdg, hosts] }

```

There are two SoftwareConfig resources:

- The first SoftwareConfig, `export_hosts`, uses the `facter` tool to extract IP address and hostname into a single line (in the format expected for `/etc/hosts`) and writes it to a special path (`${heat_outputs_path}.hosts`). This prompts Heat to assign the content of this file to the output with the name `hosts`.
- The second SoftwareConfig, `populate_hosts`, takes as input a variable named `hosts`, and applies a script that reads the variable from the environment, parses it with `ast.literal_eval` (as it is formatted as a Python dict), and writes each value of the dictionary to `/etc/hosts`.

The `SoftwareDeploymentGroup` resources `export_hosts_sdg` and `populate_hosts_sdg` apply each `SoftwareConfig` to the instance `ResourceGroup` with the correct configuration.

Finally, the instance `ResourceGroup` is configured so that each instance uses the following contextualization method instead of a `user_data` script:

```

user_data_format: SOFTWARE_CONFIG
software_config_transport: POLL_SERVER_HEAT

```

You can follow the same template pattern to configure your own deployment requiring all-to-all information exchange.

17.7.2 Automated Deployment

On Chameleon you can configure a Heat Stack to launch as soon as your lease begins. Whether your experiments require a large cluster or a single node, automated deployment saves you time configuring your environment and even allows you to run your entire experiment automatically when the necessary resources become available.

At present, you will need to use our customized versions of the Heat and Blazar CLI tools to implement this feature.

Install Custom CLI

You can install Chameleon's `python-heatclient` and `python-blazarclient` packages via `pip` by running the following commands:

```
pip install git+https://github.com/ChameleonCloud/python-heatclient.git
pip install git+https://github.com/ChameleonCloud/python-blazarclient.git
```

Initialize Stack

Next you will need to configure a Heat stack with the `--initialize` flag on the CLI and a dummy `reservation_id` parameter. The dummy id can be anything (even an empty string) so long as the `reservation_id` parameter is specified so that Blazar can overwrite it once your advanced reservation is scheduled and the stack is ready to launch. Once your stack is initialized, the status should read `INIT_COMPLETE`. This indicates that your template was validated and all the data required to launch a stack has been stored. See example command below:

```
openstack stack create -t <template_file> --initialize --parameter reservation_id=dummy
↪<stack_name>
```

Create Reservation with Stack_ID

Finally, for a stack to launch when your reservation begins, we need to let Blazar know which stack to notify Heat to update. This is done via the command line by specifying `orchestration` as an `on_start` action with a `stack_id` (e.g. `on_start=orchestration:<stack_id>`) under the `--reservation` flag. Under the hood, Blazar will update your initialized Heat stack with the `reservation_id` assigned to the lease. See example below:

```
openstack reservation lease create --start-date "<start_date>" --end-date "<end_date>" \
  --reservation min=<min>,max=<max>,resource_type=physical:host,on_start=orchestration:
↪<stack_id> \
  <lease_name>
```


OBJECT STORE

Chameleon provides an object store service through the [OpenStack Swift](#) interface. It is intended to be used for storing and retrieving data used during experiments, such as input files needed for your applications, or results produced by your experiments.

Hint: Chameleon object store service is currently backed by a [Ceph](#) cluster with more than 2.1 PB of capacity. The data is replicated, keeping two copies of each object, effectively providing over 1 PB of storage available to users. This storage capacity will increase as the project goes on. The replication should provide good availability in case of hardware failures. However, all copies are kept within the same data center and are not backed up on a separate system; if you feel that this does not provide sufficient reliability in your case, you should consider backing up really critical data externally.

18.1 Availability

You can access the *Object Store* from instances running on [CHI@TACC](#) and [CHI@UC](#). Each region has its own store, meaning that objects uploaded to one are not visible to the other. In general you should use the store local to the region where your instances are running for the best performance. To make it easier for you to use the *Object Store* client, we installed it in all appliances supported by Chameleon. Additionally, you can also access the *Object Store* from the [CHI@TACC](#) or [CHI@UC](#) web interfaces under the *Object Store* panel.

Hint: [KVM@TACC](#) users can access the TACC store by using their [CHI@TACC OpenStack RC file](#).

18.2 Objects and Containers

Objects are equivalent to individual files. They are stored in *Containers*, which are data structures that can contain multiple *Objects*. When uploading *Objects*, they must be stored inside of *Containers*. You may perform operations on individual *Objects* inside Containers, such as downloading or deleting them. You may also work with entire *Containers* and perform operations such as downloading an entire *Container*.

18.3 Managing Object Store using the GUI

To access the *Object Store* using the GUI at [CHI@TACC](#) or [CHI@UC](#), use the navigation sidebar to go to *Project > Object Store > Containers*.

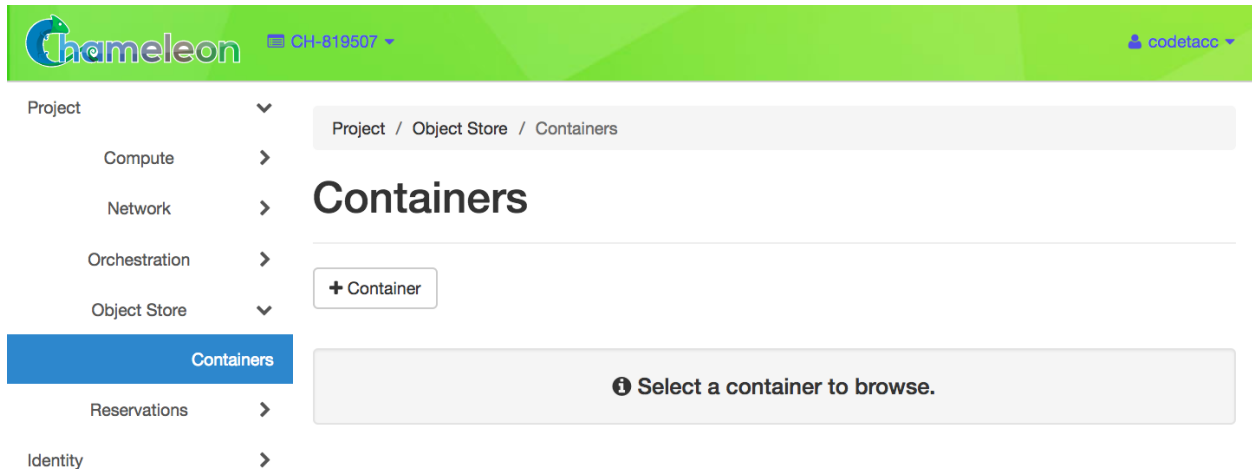


Fig. 1: The Containers page

18.3.1 Working with Containers

To create a container, click the *+Container* button. This will open the *Create Container* dialog.

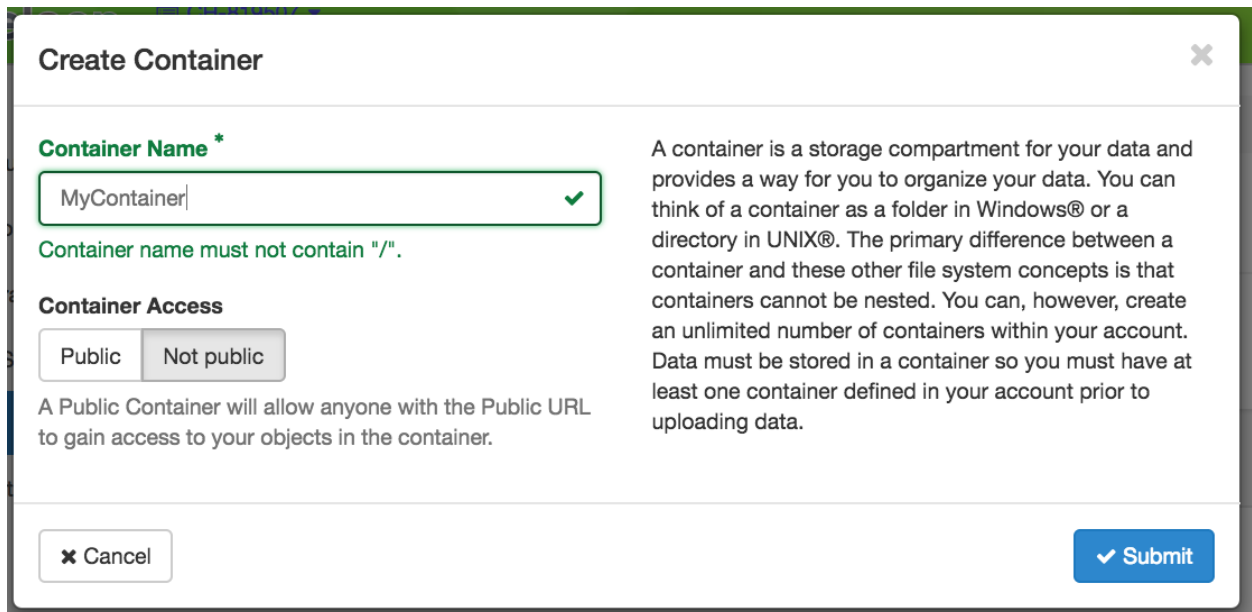


Fig. 2: The Create Container dialog

Choose a unique name of your container and set the visibility to either *Public* or *Not Public*. When you are finished, click the *Submit* button. You will see your new *Container* appear in the list on the *Containers* page.

Containers

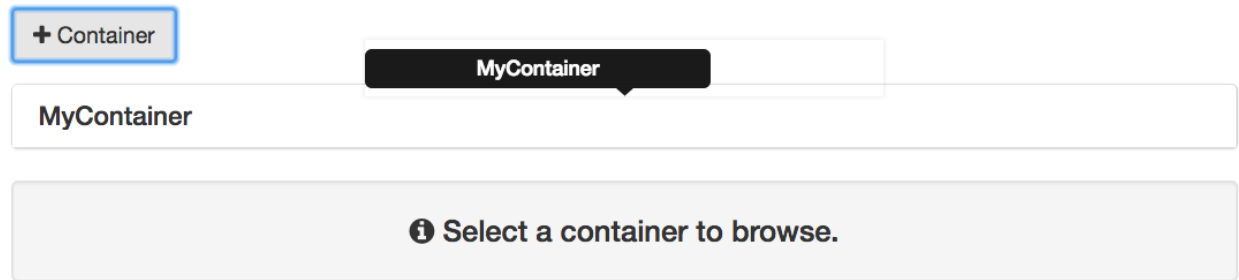


Fig. 3: The Container list

You may click on a *Container* to see the details and work with *Objects* belong to it.

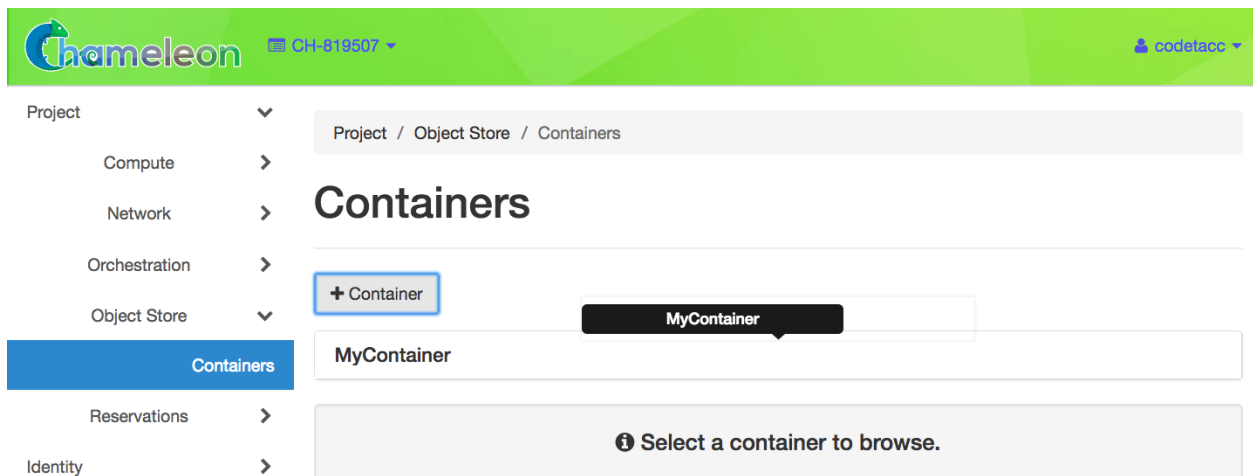


Fig. 4: Container details

Attention: Downloading a container is not available from the GUI. Use the CLI to download containers.

You may delete a container by clicking the *Delete* icon in the upper right of the *Container Detail Panel*.

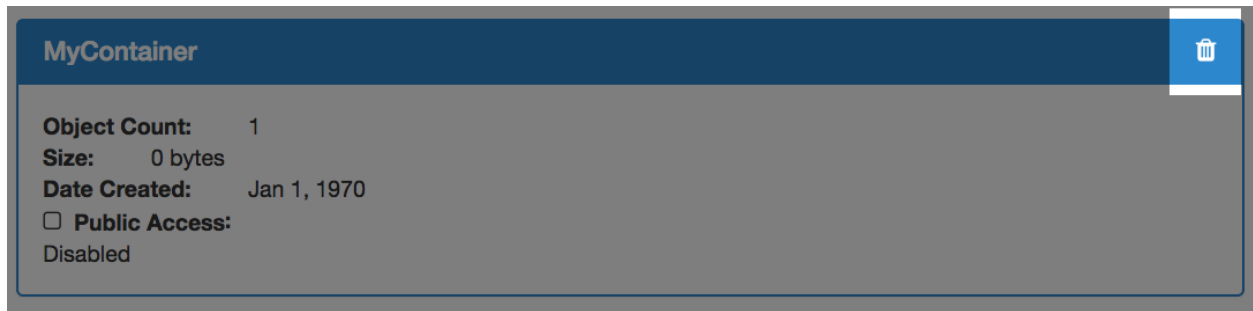


Fig. 5: The Delete Container button

18.3.2 Working with Objects

To upload a local file to a container, click the button with the *Upload* symbol next to the search bar.

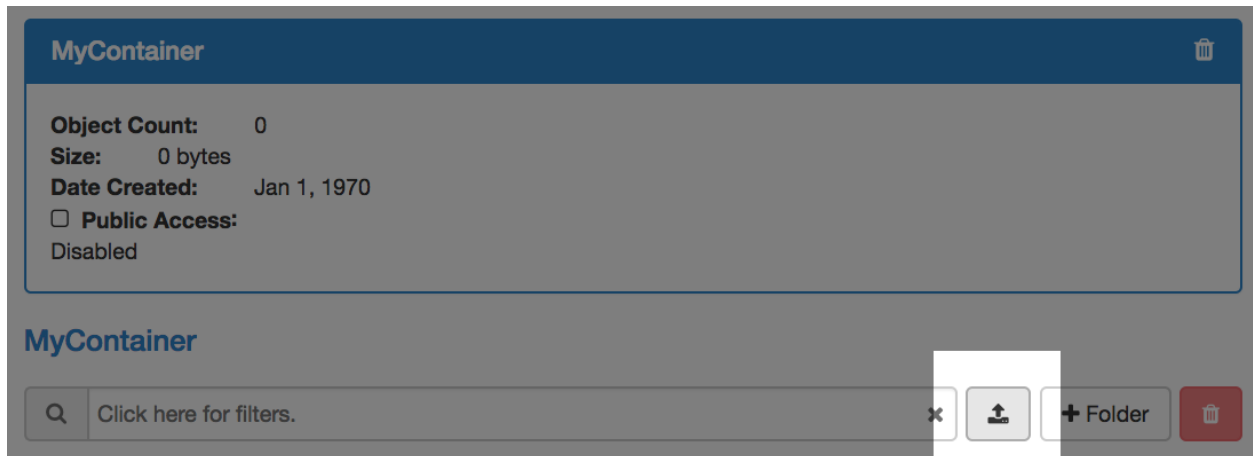


Fig. 6: The Upload button

This will open the *Upload File* dialog.

Choose a file to upload from your local file system and give a name to the object.

18.3.3 Working with Folders

If you wish to create a *Folder* within your *Container*, click the *+Folder* button and give a name to your folder in the *Create Folder* dialog.

Your new folder will appear in the *Container details*.

You may browse your folder and upload files to it by clicking on the folder.

Upload File To: MyContainer

File
 No file chosen

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

Fig. 7: The Upload File dialog

Create Folder In: MyContainer

Folder Name

Note: Delimiters ('/') are allowed in the folder name to create deep folders.

Fig. 8: The Create Folder dialog

MyContainer

Click here for filters.

Displaying 1 item

<input type="checkbox"/> Name ^	Size
<input type="checkbox"/> MyFolder	Folder

Displaying 1 item

Fig. 9: A Container with a Folder

MyContainer : MyFolder

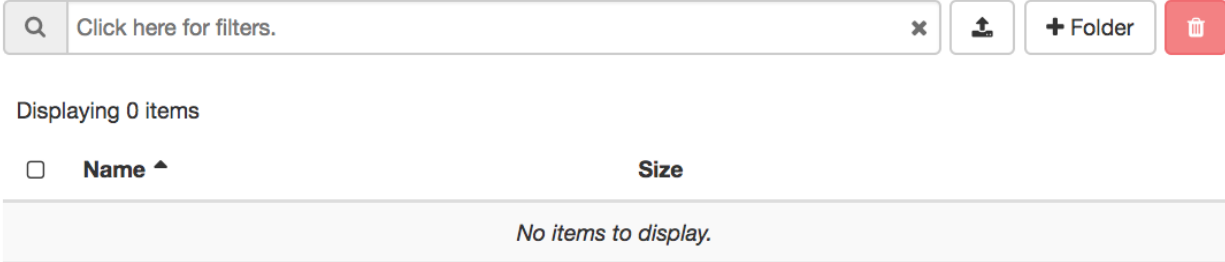


Fig. 10: A Folder within the Container

18.4 Managing Object Store using the CLI

Tip: Reading *Command Line Interface (CLI)* is highly recommended before continuing on the following sections.

In addition to *Installing the CLI*, you must also install `python-swiftclient` package:

```
pip install python-swiftclient
```

Then, you must set environment variables for your account and project using *The OpenStack RC Script*.

18.4.1 Working with Containers

To create a *Container*, use the following command:

```
openstack container create <container_name>
```

Tip: By default, the *Container* created using the above command will not be visible to the public.

To view all containers that belong to your project, run:

```
openstack container list
```

Tip: You may use `--prefix <prefix>` as a filter to list the containers whose name starts with `<prefix>`.

To see details of a container, use the command:

```
openstack container show <container_name>
```

To view a list of objects within a container, use the command:

```
openstack object list <container_name>
```

To download a container with all the objects belong to it, use the following command:

```
openstack container save <container_name>
```

To delete a container and wipe out all the objects belong to it, use the following command, and **be careful!**

```
openstack container delete --recursive <container_name>
```

18.4.2 Working with Objects

You may upload a file from your local machine to a container using the following command:

```
openstack object create <container_name> <local_filename>
```

Tip: Optionally, you may name the object differently from it's original name in your local machine by using the `--name` parameter.

To delete an object from a container, run:

```
openstack object delete <container_name> <object_name>
```

If you wish to download an individual object directly from a container, use the command:

```
openstack object save <container_name> <object_name>
```

Large object support

The Swift CLI only supports objects up to 4GB. Larger objects are supported, provided they are uploaded in segments. This advanced functionality is only supported using a separate Swift interface. For a version compatible with Chameleon's authentication, you need *python-swiftclient* `>= 3.11.1`, and to generate and use an *Application Credential*

```
pip install "python-swiftclient>=3.11.1"
```

Instead of invoking commands via `openstack`, you will instead use the `swift` command, which supports a `--segment-size` parameter, specifying the segment size in bits. `--segment-size 4831838208` is close to the segment limit of 4GB.

There is also a `--changed` flag, which prevents uploading of the object if the checksum has not changed:

```
swift --os-auth-type v3applicationcredential \
--os-application-credential-id <credential_id> \
--os-application-credential-secret <credential_secret> \
upload --changed --segment-size 4831838208 \
<container_name> <path>
```

18.4.3 Working with Folders

There isn't "folders" when you managing the *Object Store* with the CLI. However, when you create an object, you may use the delimiter / to specify the path.

18.5 Managing Chameleon Cloud Object Store with AWS CLI

In addition to the Openstack Swift API, Chameleon's object store can be accessed via an S3 compatible API. Although we don't directly support them, you can use most s3 compatible clients Chameleon, so long as they allow you to set the endpoint, as well as the "path-style" access url.

As an example, this sections provides instructions on configuring the AWS CLI to connect to our object store.

18.5.1 Prerequisites

Before you begin, make sure you have the AWS CLI installed on your local machine. You can download it from the [official AWS CLI website](#).

18.5.2 Configuration

1. Obtain Chameleon Cloud Object Store Access Credentials:

To use the AWS CLI with Chameleon Cloud Object Store, you need to obtain your access credentials using the following command:

```
openstack ec2 credential create
```

This command will provide you with an Access Key ID and Secret Access Key, which you will use to configure the AWS CLI.

2. Configure AWS CLI:

Run the following command to configure the AWS CLI with the obtained credentials:

```
aws configure
```

You will be prompted to enter the Access Key ID, Secret Access Key, default region, and output format. Enter the values accordingly.

Example:

```
AWS Access Key ID [None]: <your-access-id-from-openstack-ec2-create>
AWS Secret Access Key [None]: <your-secret-from-openstack-ec2-create>
Default region name [None]: <region-can-be-anything>
Default output format [None]: json
```

3. Set Endpoint for Chameleon Cloud Object Store:

The endpoint for Chameleon Cloud Object Store is:

- For UC: <https://chi.uc.chameleoncloud.org:7480>
- For TACC: <https://chi.tacc.chameleoncloud.org:7480>

Run the following command to set the endpoint:


```
aws configure set endpoint_url <endpoint-url>
```

Replace <endpoint-url> with the appropriate endpoint based on your Chameleon Cloud site.

18.5.3 Usage

Once configured, you can use the AWS CLI commands to interact with Chameleon Cloud Object Store as you would with any S3-compatible storage service.

Example:

```
# List S3 buckets
aws s3 ls

# Upload a file to a bucket
aws s3 cp local-file.txt s3://<your-bucket>/
```

Replace <endpoint-url> with the appropriate bucket name.

That's it! You have successfully configured the AWS CLI to work with Chameleon Cloud Object Store. For more information [this article](#) should help

18.6 Mounting Object Store as a File System

Tip: Cloudfuse can upload objects up to 4GB. For larger objects, please use the Swift CLI.

When logged into an instance using Chameleon-supported images, such as [CC-CentOS8](#) and [CC-Ubuntu18.04](#), you will see a directory called `my_mounting_point` which is a pre-mounted directory to your Chameleon Object Store at the same site of your instance. Each Object Store container that you have access to will appear as a subdirectory inside this mount.

The `cc-cloudfuse` tool (Source: [ChameleonCloud/cc-cloudfuse](#)) is pre-installed in Chameleon-supported images. It is based on the `cloudfuse` tool (Source: [redbo/cloudfuse](#)), which is used to mount your Chameleon Object Store as a directory on your Linux environment.

Important: Some older Chameleon-supported images have an outdated version of this tool installed, which is not compatible with authentication using federated identity. If you wish to continue using a historical image, you should update the tool by following the [installation instructions](#).

To mount, use the following command:

```
cc-cloudfuse mount <mount_dir>
```

Now you can access your Chameleon Object Store as your local file system.

To unmount, use the following command:

```
cc-cloudfuse unmount <mount_dir>
```

Important: Limitations

The primary usage scenario of the `cc-cloudfuse` tool is to allow you to interact with Chameleon Object Store using familiar file system operations. Because the `cc-cloudfuse` runs on top of an object store, it is important to understand that not all functionality will behave identically to a regular file system.

1. Symbolic links, file permissions, and POSIX file locking operations are not supported.
2. Updating an existing file is an expensive operation as it downloads the entire file to local disk before it can modify the contents.
3. You can mount from multiple nodes, but there is no synchronization between nodes regarding writes to Object Storage.
4. The mounting root directory can only contain directories, as they are mapped to Object Store containers.
5. Renaming directories is not allowed.
6. It keeps an in-memory cache of the directory structure, so it may not be usable for large file systems. In addition, files added by other applications will not show up until the cache expires.
7. The maximum number of listings is 10,000 items.

Please keep these limitations in mind when evaluating `cc-cloudfuse`.

Note: You may experience persistence issues when using `cc-cloudfuse`, especially when writing large files or writing many files at the same time. Unmounting and re-mounting usually resolves this.

SHARES

Chameleon provides a shared file system service through the [OpenStack Manila](#) interface. With the service, you can create a shared file system, mount to the bare metal instances, and manage some of its properties, such as visibility.

Hint: Chameleon shared file system service is currently backed by a CephFS. Same as our [object store](#) service, the data is replicated and the replication should provide good availability in case of hardware failures.

Difference between shared file system and object store

You can choose either shared file system or object store to store, manage, and share your data with your collaborators. The object store is suitable for storing large objects at scale, but isn't suitable for transactional data, as objects are immutable and updated in their entirety. Chameleon shared file system instead provides a NFS mount to the bare metal instances, with the NFS protocol managing locking and data integrity processes required to provide multiple concurrent access to data.

The shared file system service is available at [CHI@UC](#) and [CHI@TACC](#). Each region has its own service and the shares created at one region are not available to the other. As all other Chameleon services, you can create and manage your shares using both GUI and CLI.

19.1 Storage Networks

To provide isolation among shares created by different projects, accessing a share requires a storage network, which are special networks you can reserve to use. When reserving a storage network, add `usage_type=storage` to the resource properties. To learn more about reserving networks, read the [reservations documentation](#). All bare metal instances that are created on the storage network have access to all the project shares.

Tip: To attach floating IP to your instance created on a storage network, you need to create a router with *public* external network. Then connect the storage subnet to the router. You must specify an unused IP address which belongs to the selected subnet. To learn more about creating router and connecting subnet, please read [isolated network VLANs](#).

19.2 Shares

19.2.1 Visibility

Shares are owned by the project. By default, all shares have *private* visibility and can only be listed and accessed within your project. All bare metal instances owned by the project have read and write permissions to the project's shares. You can also make your shares *public*. All Chameleon users and projects can list public shares, and with a storage network, all projects have read-only access to a public share.

19.2.2 Accessibility

A share is a pre-allocated storage space at a CephFS. You can *mount your shares to your bare metal instances via NFS protocol*. The accessibility of the shares are controlled internally by the reservation service. You are not allowed to edit the access rules of a share.

19.2.3 Quotas

We do not charge SUs for the storage spaces of your shares. However, we do limit the total size and the number of shares you can create within your project. The maximum number of shares is 10 and the maximum size allowed for all shares in a project is 2000 GiB. If you need to increase the default quota, please submit a ticket via the [Help Desk](#).

19.3 Managing Shares using GUI

To manage your share, use the *Shares* page at [CHI@UC](#) or [CHI@TACC](#) by navigating to *Project > Share > Shares*.

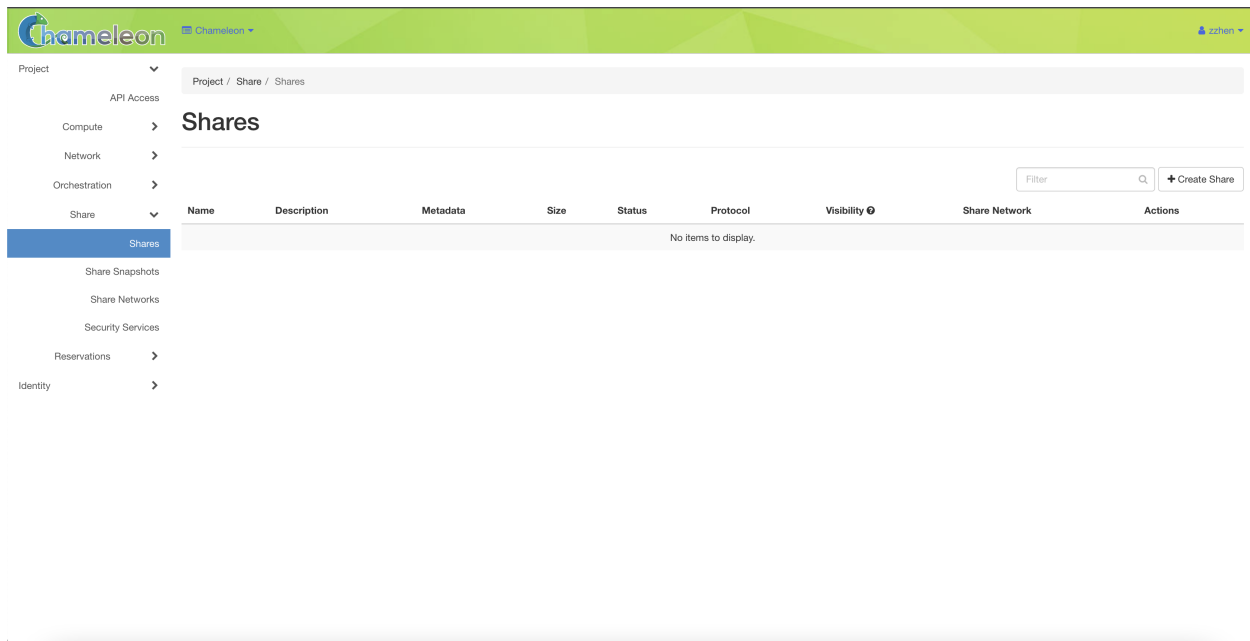


Fig. 1: The Shares page

19.3.1 Create Share

Click the *Create Share* button. In the *Create Share* dialog, provide a name and the size of your share, and then click the *Create* button to create a share.

Note: A storage network is not required for creating shares. It's only required to access the shares.

19.3.2 View Share

You can look at the details of a share by clicking the share name in the *Shares* page. Note that the paths of the *export locations* are important as you will use this path to mount your share to your bare metal instances. You can also see the other properties, such as visibility and size. The access rules are listed in the *share details* page, though you can not edit the rules, as they are controlled by the reservation service.

19.3.3 Edit Share

You can manage the properties and extend the size of a share by clicking the *Action* dropdown in the *Shares* page.

19.3.4 Delete Share

You can use the *Action* dropdown to delete a single share, or select multiple shares and click the *Delete Shares* button.

Important: Be careful when deleting shares, as the action is irreversible. However, the termination of your storage network reservation **DOES NOT** delete your share. Your shares persist until you manually delete them.

19.4 Managing Shares using CLI

As all other Chameleon services, you can manage your shares via CLI as well.

Tip: Reading *Command Line Interface (CLI)* is highly recommended before continuing on the following sections.

In addition to installing the CLI, you must also install *python-manilaclient* package:

```
pip install python-manilaclient
```

Then, you must set environment variables for your account and project using *The OpenStack RC Script*.

Tip: If you get HTTP 406 error of `version is not supported by the API`, please add `--os-share-api-version 2.65` to the command to specify manila minor version.

Create Share

Share Name *

demo

Description

Share Protocol *

NFS

Size (GiB) *

1

Share Type *

cephfsnfstype

Availability Zone

Metadata

☐ **Make visible for all** ?

Description:

Select parameters of share you want to create.

Metadata:

One line - one action. Empty strings will be ignored.
To add metadata use:

key=value

Share Limits

Total Gibibytes

0 of 1,000 GiB Used

Number of Shares

0 of 50 Used

Cancel

Create

Fig. 2: The Create Share dialog

Project

API Access

Compute

Network

Orchestration

Share

Shares

Share Snapshots

Share Networks

Security Services

Reservations

Identity

Project / Share / Shares / Share Details: demo

Share Details: demo

Share Overview

Name

ID

Status

demo

f01699e9-fe01-4b93-9c2f-9606fca0b965

Available

Export locations

Path: 10.140.83.254:/volumes/_nogroup/e09b8972-3dd6-46e4-be10-2930ac26149b/f181cf0a-d567-4e05-8f6d-2461ec33f6be

Preferred: False

Visibility

Availability zone

Size

Protocol

Share type

Name

ID

Mount snapshot support

Created

Task state

private

nova

1 GiB

NFS

cephfsnfs

demo

baced923-0cd0-43a6-a2d9-7eb8857527c3

False

May 18, 2022, 7:08 p.m.

None

Access Rules

Metadata

Fig. 3: The Share details

Shares

Filter

+ Create Share

Delete Shares

Displaying 1 item

<input type="checkbox"/>	Name	Description	Metadata	Size	Status	Protocol	Visibility	Share Network	Actions
<input type="checkbox"/>	demo	-		1GiB	Available	NFS	private	-	<div>Edit Share</div> <div><div>Extend Share</div><div>Manage Rules</div><div>Edit Share Metadata</div><div>Delete Share</div></div>

Displaying 1 item

Fig. 4: The Action dropdown

19.4.1 List Shares

To list all shares of your project, run the following command:

```
openstack share list
```

You can filter the results by the share name via adding a `--name` argument to the list command.

19.4.2 Create Share

To create a share, using the following command:

```
openstack share create --name <name of your share> NFS <size in GiB>
```

For example, for creating a 1 GiB share with name of `my-first-share`, run:

```
openstack share create --name my-first-share NFS 1
```

Note: Only the NFS protocol is supported.

You can add the `--public true` to make your share public.

19.4.3 Edit Share

To change the visibility of a share, run:

```
openstack share set --public <true/false> <name/id of the share>
```

To update the name or the description of a share, run:

```
openstack share set --name <new name> --description <description> <name/id of the share>
```

To extend/shrink the size of a share, run:

```
openstack share resize <name/id of the share> <new size in GiB>
```

19.4.4 View Share

To view the details of a share, run:

```
openstack share show <name/id of the share>
```


19.4.5 Delete Share

To delete a share, run the following command:

```
openstack share delete <name/id of the share>
```

19.5 Mounting Shares to Instances

In order to allow your instances to access the share, you need to create your instances using the *pre-reserved storage network*. To learn more about how to create a bare metal instance on a network, read *the bare metal instances section*.

Important: The shares are independent of the storage networks. You can create shares any time regardless of the status of the storage networks. The storage networks are only used to access your data stored in the share.

After your instance becomes active, find the export location path of the share using *GUI* or *CLI*. To mount the share, run the following command:

```
sudo mount -t nfs -o nfsvers=4.2,proto=tcp <export location path> <mount dir>
```

Now, you can read and write to the share and it behaves identically to a regular file system.

To unmount, run the following command:

```
sudo umount <mount dir>
```


NETWORKING

Networking on Chameleon is implemented using [OpenStack Neutron](#). Most experiments will require *Basic Networking* functionality including Internet access and connectivity between nodes. Chameleon provides basic networking capabilities via a pre-configured shared network called `sharednet1`. Many experiments require additional connectivity and control of the network. These experiments can utilize Chameleon’s advanced networking capabilities including *Isolated Network VLANs*, *External Layer2 Connections (Stitching)*, and *Software Defined Networking*.

20.1 Basic Networking

Note: Step-by-step instructions for getting started with Chameleon are available in the *Getting started* section of this documentation. These instructions include using basic networking functionality.

20.1.1 Shared Network

All Chameleon Projects have access to the fixed network `sharednet1` which is used by most experiments. The `sharednet1` is a pre-configured network shared among all Chameleon Projects with one *Subnet* and includes a *Router* providing NAT access to the public Internet. All instances using `sharednet1` can communicate directly.

20.1.2 Multiple Networks

Some Chameleon bare metal nodes support connecting to multiple networks. Currently, the number of networks allowed is limited to the number of enabled NICs on the node (currently this is up to 2). It is possible to find such nodes via *Resource discovery* by filtering by the “Enabled” flag for a given Network Adapter slot. Note that the slots are 0-indexed, meaning the first NIC is referred to as Network Adapter #0.

When launching a node that supports multiple networks, simply assign multiple networks to the instance when you are launching it. The networks will be mounted on NICs in the same order that the networks are assigned; that is, the first assigned network will be mounted on Network Adapter #0, and the second on Network Adapter #1, and so on.

20.1.3 Floating IP Addresses

Instances on Chameleon are assigned a *fixed* IP address that can be used for local connectivity as well as NAT access to the public Internet. A publicly accessible IPv4 address (*Floating IP address*) is required in order to access Chameleon instances from the Internet or host public services. [CHI@TACC](#) and [CHI@UC](#) each have a limited number of public IP addresses that can be allocated to your instances.

The [Getting started](#) guide shows how to allocate *Floating IP address* to your nodes.

Important: The Chameleon floating IP address pool is a shared and finite resource. **Please be responsible and release any unused floating IP address, so other Chameleon users and projects can use them!**

Floating DNS Records

Each Floating IP is also contained within a dedicated DNS A record; this means that you can access your instance over the Internet either via its Floating IP or a special hostname. This can be particularly helpful if you want to set up a TLS certificate for HTTPS to secure a service you are exposing over the web, e.g., with [LetsEncrypt](#).

Site	Hostname pattern (for IP <i>AA.BB.CC.DD</i>)
CHI@TACC	chi-dyn-AA-BB-CC-DD.tacc.chameleoncloud.org <i>e.g.,</i> <i>chi-dyn-129-114-108-147.tacc.chameleoncloud.org</i>
CHI@UC	chi-dyn-AA-BB-CC-DD.uc.chameleoncloud.org <i>e.g.,</i> <i>chi-dyn-192-5-87-98.uc.chameleoncloud.org</i>

20.1.4 Security

When your instance has a *Floating IP address* assigned, it is reachable directly over the public Internet. For this reason, it is important to consider the security of any services running on your instance. In particular, **ensure that you have not allowed SSH authentication with passwords** (this is disabled by default on Chameleon-supported images.)

In order to better protect Chameleon instances, Chameleon Ubuntu and CentOS base images come with baked-in firewall rules which severely restrict connections over the public internet. If using a different image or if you disable firewall rules, realize that any network services can potentially be exposed to the public Internet if your instance has a Floating IP attached.

Warning: Some commodity systems such as Apache Spark and Hadoop have in the past shipped with *very insecure* default settings. Pay particular attention to the security needs of your experiment when selecting what systems you need to install on your node, particularly when exposing the node to the Internet.

Note: We're here to help! If you want advice on how to securely run your experiment, feel free to file a [Help Desk](#) ticket.

Firewall

Chameleon-supported Ubuntu and CentOS images are preconfigured with a firewall utility called `firewalld` enabled and the following rules set:

```
# sudo firewall-cmd --zone=public --list-services
dhcpv6-client ssh
```

These rules allow ssh traffic on port 22 over the public internet.

Warning: By default, all firewall changes are **temporary**, and will be lost on instance reboot. This is a safety mechanism to avoid locking yourself out. To make changes **permanent**, execute:

```
sudo firewall-cmd --runtime-to-permanent
sudo firewall-cmd --reload
```

To enable HTTP/HTTPS on port 80 and 443:

```
sudo firewall-cmd --zone=public --add-service http
sudo firewall-cmd --zone=public --add-service https
```

Firewalld has many “built-in” rules for common services, but you can also enable communication over a specific port using the command:

```
# list all open ports
sudo firewall-cmd --zone=public --list-ports

# open a new port
sudo firewall-cmd --zone=public --add-port=<port>/<protocol>

# example
sudo firewall-cmd --zone=public --add-port=9001/tcp
```

You can also permit connections from a specific ip or network, such as a trusted endpoint, or within your own isolated networks on Chameleon.

```
sudo firewall-cmd --zone=trusted --add-source=<your_subnet_cidr/netmask>
```

To enable this by default for all private IP ranges, you can do the following, but please note that this can be insecure on shared or routed networks (`sharednet1`, `sharedwan1` and similar).

```
sudo firewall-cmd --zone=trusted --add-source=192.168.0.0/16
sudo firewall-cmd --zone=trusted --add-source=172.16.0.0/12
sudo firewall-cmd --zone=trusted --add-source=10.0.0.0/8
```

Any other incoming connections will be denied.

For more examples and information, please see:

- [Ubuntu’s man page for firewalld](#)
- [Fedora Linux Guide](#)
- [Rocky Linux Guide](#)

Security Groups

KVM@TACC supports *Security Groups*, which can be assigned directly to instances upon launch or after the instance is already running. By default, instances have no *Security Groups* applied, so all traffic is allowed.

Limit bound interfaces

Instead of binding a web service to all interfaces (e.g. `0.0.0.0` for IPv4, `::` for IPv6), consider listening only on the node's private IP, which is not routable from the public Internet. If you can, listening on localhost (`127.0.0.1`) is even safer. Most web services have a way to specify the bind address and some default to binding on all interfaces, which is often insecure.

20.2 Isolated Network VLANs

By default, bare metal nodes on each Chameleon site share the same local network (shared VLAN and IP subnet). However, some experiments may require more network isolation, which is now supported by Chameleon.

Chameleon's implementation of network isolation is based on dynamically managed VLANs (network layer 2) associated with user-configured private IP subnets (network layer 3). This means that all network communications local to the IP subnet or the broadcast domain (such as Ethernet broadcast, ARP, IP broadcast, DHCP, etc.) will be restricted to the user-configured network and its associated VLAN. This feature enables a range of experiments in networking and security. For example, this allows running your own DHCP server to configure virtual machines running on bare metal nodes, without impacting other users.

Note:

- Strong network isolation is provided at network layer 2 only. Even using separate IP subnetworks, any bare metal node can still communicate with each other and with the Internet through the network's router. We are investigating solutions to provide stronger isolation at network layer 3.
- Network isolation works on all nodes, including our low-power HP Moonshot nodes (low-power Xeon, Atom, ARM64).

To use this feature, you will need to create a dedicated network and router. You can use a [Heat template](#), use the *Network* panel of the GUI, or use the CLI.

20.2.1 Configuring Networking using a Heat template

1. Go to *Project > Orchestration > Stacks*.
2. Click the *Launch Stack* button to open an interactive dialog.
3. Select *URL* as *Template Source* and paste <https://raw.githubusercontent.com/ChameleonCloud/heat-templates/master/network-isolation/network-isolation.yaml> to *Template URL*.
4. Click the *Next* button to navigate to the *Launch Stack* dialog.
5. Provide a name for your stack, enter your password, and set a private IP range, such as `192.168.1.0/24`.
6. Set the first and the last IP addresses of *DHCP* range.

Important: The first IP address in the DHCP range should never be *.1 and *.2. The last IP address in the range must be less than *.255.

7. Start creating the network and router by clicking the *Launch* button.

20.2.2 Creating a Network using the GUI

To create a Network from either the *Network Topology* page or the *Networks* page, click the +*Create Network* button to open the *Create Network* dialog.

Fig. 1: The Create Network dialog

In *Create Network* dialog, name your network. In general, you will also want to create a *Subnet* for your new Network, so make sure you have *Create Subnet* checked. Click the *Next* button.

When creating a *Subnet*, you must specify a *Subnet Name* and a *CIDR Network Address* that contains a private IP address and a subnet mask length. For example, you may create a *Class C* subnet with a 24-bit mask by entering 192.168.1.0/24. You may set a Gateway or leave it blank to use the default. Then, click the *Next* button.

Attention: Do not select the *Disable Gateway* checkbox!

You may specify *DHCP* and static *Route* information at *Subnet Details* section:

- *Allocation Pools* section allows you to specify *DHCP* address ranges in the format of <first address>, <last address>. For example, entering 192.168.1.2, 192.168.1.100 will create a *Subnet* with IP ranges from 192.168.1.2 to 192.168.1.100.
- *DNS Name Servers* section allows you to specify a list of DNS servers.

Note: At CHI@TACC, use 129.114.97.1 and 129.114.97.2 for your DNS servers At CHI@UC, use 8.8.

Create Network

[Network](#)[Subnet](#)[Subnet Details](#)

Subnet Name

Network Address ?

IP Version

Gateway IP ?

☐ **Disable Gateway**

Creates a subnet associated with the network. You need to enter a valid "Network Address" and "Gateway IP". If you did not enter the "Gateway IP", the first value of a network will be assigned by default. If you do not want gateway please check the "Disable Gateway" checkbox. Advanced configuration is available by clicking on the "Subnet Details" tab.

Cancel

« Back

Next »

Fig. 2: The Subnet tab

Create Network

[Network](#)[Subnet](#)[Subnet Details](#)

☒ **Enable DHCP**

Specify additional attributes for the subnet.

Allocation Pools ?

DNS Name Servers ?

Host Routes ?

Cancel« BackCreate

Fig. 3: Subnet details

8.8 and 8.8.4.4 for your DNS servers

- *Host Routes* section allows you to specify static routing information for the subnet in the format of <subnet CIDR>, <router IP address>. For example, 192.168.3.0/24, 10.56.1.254 means all traffic from this Subnet to 192.168.3.0 will be forwarded to the Router Interface at 10.56.1.254.

Note: All three sections above are line separated.

Click *Create* button and a new Network will be created. Check if the network is created without error.

Creating a Router

To create a *Router* from either the *Network Topology* page or the *Routers* page, click the +*Create Router* button to open the *Create Router* dialog.

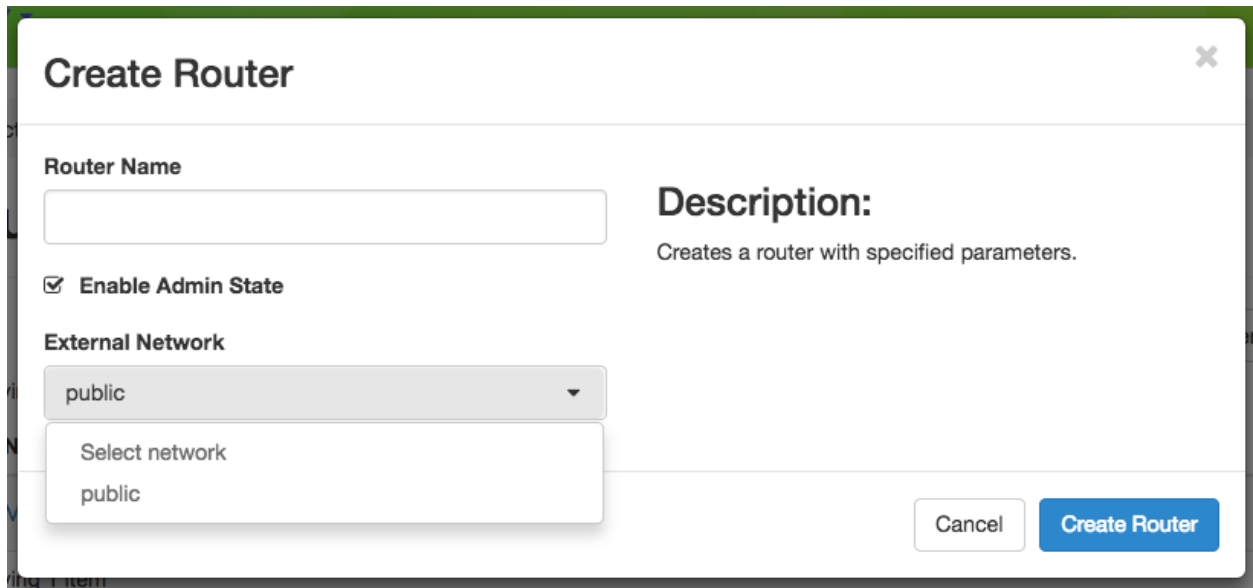


Fig. 4: The Create Router dialog

In this dialog, specify a *Router Name*. Optionally, you may select *public* as the *External Network* if you want to have external access. Click *Create Router* to complete the process.

Adding a Router Interface

A Router may have multiple *Interfaces*, each connected to a *Network*. You may add an *Interface* to an existing *Router* by clicking on *Add Interface* from either the *Network Topology* page or the *Routers* page to open the *Add Interface* dialog.

First, select a network and subnet you have created. You can specify an *IP address*; otherwise, Chameleon will attempt to assign an IP address automatically. The gateway IP you assigned to the subnet will be automatically picked.

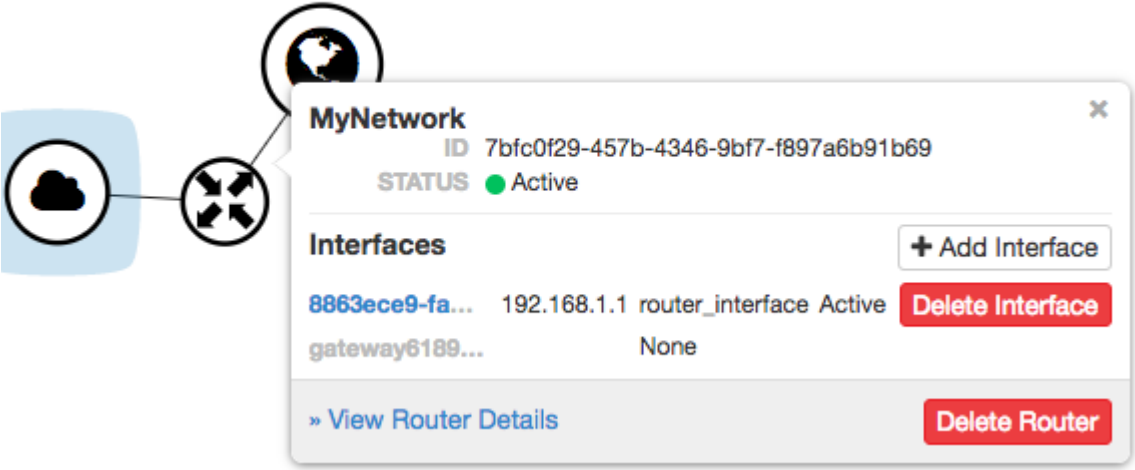


Fig. 5: The Router interface in the Network Topology page

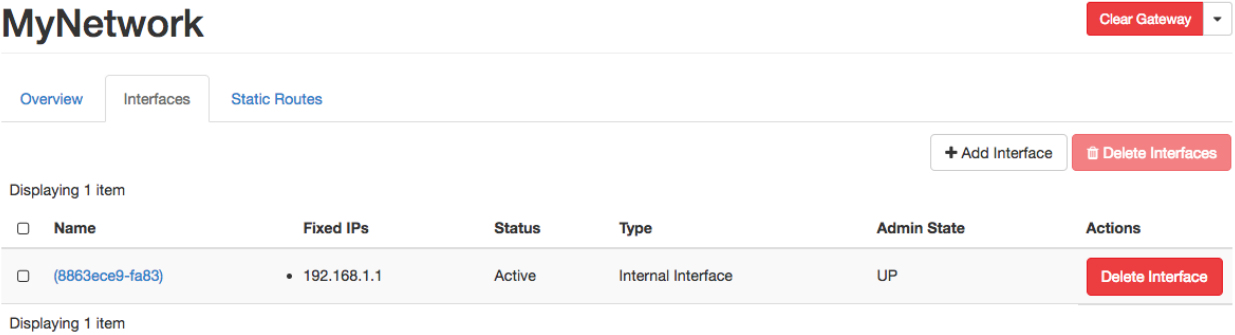
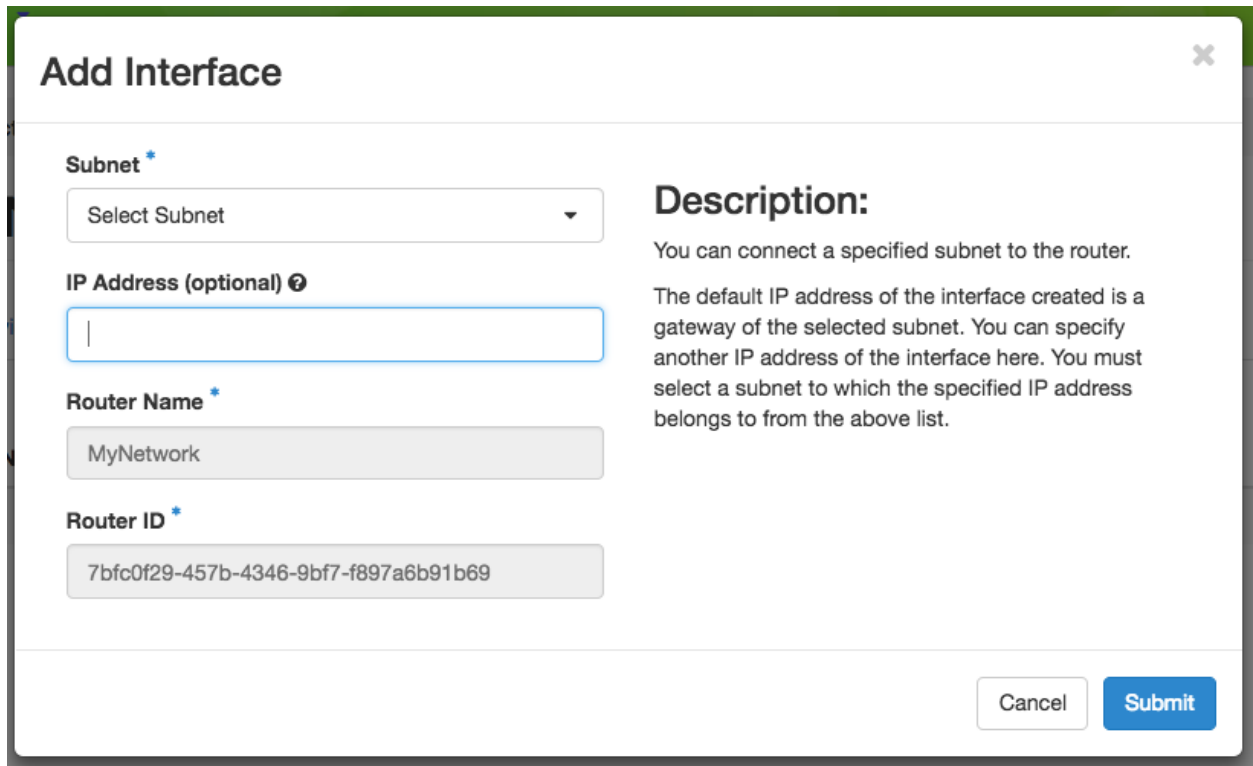


Fig. 6: The Interfaces tab in the Router detail page



The image shows a modal dialog titled "Add Interface" with a close button (X) in the top right corner. The dialog is divided into two main sections. On the left, there are four input fields: "Subnet" (a dropdown menu with "Select Subnet" as the placeholder), "IP Address (optional)" (a text input field with a question mark icon), "Router Name" (a text input field with "MyNetwork" as the value), and "Router ID" (a text input field with a long alphanumeric string "7bfc0f29-457b-4346-9bf7-f897a6b91b69" as the value). On the right, there is a "Description:" section with two paragraphs of text. At the bottom right of the dialog, there are two buttons: "Cancel" and "Submit".

Add Interface

Subnet *

Select Subnet

IP Address (optional) ?

Router Name *

MyNetwork

Router ID *

7bfc0f29-457b-4346-9bf7-f897a6b91b69

Description:

You can connect a specified subnet to the router.

The default IP address of the interface created is a gateway of the selected subnet. You can specify another IP address of the interface here. You must select a subnet to which the specified IP address belongs to from the above list.

Cancel Submit

Fig. 7: The Add Interface dialog

Deleting Networking Objects

Attention: Network objects such as *Routers* and *Networks* must be deleted in the reverse order of which they were created. Objects **can not** be deleted while other objects are depending on them.

Attention: Before starting to delete network objects, make sure all instances using them are terminated!

1. Go to *Project > Network > Routers*, and click on the router you would like to delete.
2. Go to *Static Routes* tab, and click on the *Delete Static Routes* button in the *Action* column. The *Static Routes* will be deleted after confirm.
3. Go to *Instances* tab, delete the Gateway interface by clicking on *Delete Interface* button in the *Action* column and confirm the deletion.
4. Now you can safely delete the router by clicking on the dropdown on the upper right corner. Then, click on *Delete Router*. Finally, confirm your deletion of the router.
5. Go to *Project > Network > Networks*, and delete the network by using the dropdown in the *Action* column. Alternatively, you may delete the network by selecting the network using the checkbox and click on *Delete Networks* button on the upper right corner. Confirm your deletion to finish the process.

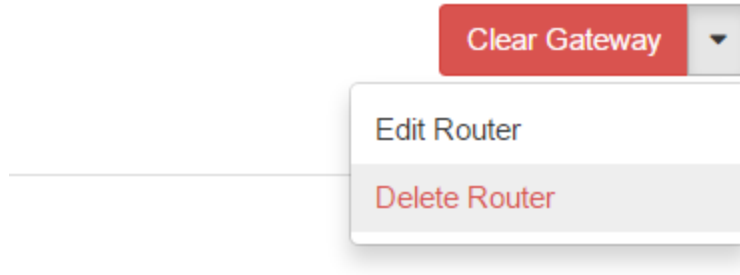


Fig. 8: Dropdown for deleting router

20.2.3 Configuring Networking using the CLI

Tip: Reading *Command Line Interface (CLI)* is highly recommended before continuing on the following sections.

Before using the CLI, make sure you have configured environment variables using *The OpenStack RC Script*.

Creating a Network

You can create an *Isolated* VLAN Network using the command:

```
openstack network create --provider-network-type vlan --provider-physical-network_
↪ physnet1 <network_name>
```

The output should look like the following:

Field	Value
admin_state_up	UP
availability_zone_hints	
availability_zones	
created_at	2018-03-23T23:45:19Z
description	
dns_domain	None
id	21ed933c-323d-4708-930c-d5f82c507430
ipv4_address_scope	None
ipv6_address_scope	None
is_default	None
is_vlan_transparent	None
mtu	1500
name	MyNetwork
port_security_enabled	False
project_id	d5233415ee0b467baec14cbd2d0e1331
provider:network_type	vlan
provider:physical_network	physnet1
provider:segmentation_id	2018

(continues on next page)

(continued from previous page)

qos_policy_id	None	
revision_number	2	
router:external	Internal	
segments	None	
shared	False	
status	ACTIVE	
subnets		
tags		
updated_at	2018-03-23T23:45:19Z	

Note: Note the provider:segmentation_id field in the above output. Each *Isolated* VLAN Network requires a unique network segment to operate. There are a finite number of valid network segments on Chameleon. If you are unable to create a network because there are no valid network segments available, then you can create a network automatically by *Creating a Lease to Reserve a VLAN Segment*.

Once you have created a Network, you may create a subnet with the command:

```
openstack subnet create --subnet-range <cidr> --dhcp --network <network_name> <subnet_
↪name>
```

For example, the command:

```
openstack subnet create --subnet-range 192.168.1.0/24 --dhcp --network MyNetwork MySubnet
```

will create a subnet with the following output:

Field	Value	
allocation_pools	192.168.1.2-192.168.1.254	
cidr	192.168.1.0/24	
created_at	2018-03-23T23:50:11Z	
description		
dns_nameservers		
enable_dhcp	True	
gateway_ip	192.168.1.1	
host_routes		
id	8be4e80d-ba49-4cdc-8480-ba43dd4724c2	
ip_version	4	
ipv6_address_mode	None	
ipv6_ra_mode	None	
name	MySubnet	
network_id	21ed933c-323d-4708-930c-d5f82c507430	
project_id	d5233415ee0b467baec14cbd2d0e1331	
revision_number	2	
segment_id	None	
service_types		
subnetpool_id	None	
tags		
updated_at	2018-03-23T23:50:11Z	

(continues on next page)

(continued from previous page)

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

To see more options when creating a subnet, use the following command:

```
openstack subnet create --help
```

Creating a Router

To create a router, use the following command:

```
openstack router create <router_name>
```

Your output should look like:

```
+-----+-----+
| Field                | Value                                |
+-----+-----+
| admin_state_up       | UP                                  |
| availability_zone_hints |                                     |
| availability_zones    |                                     |
| created_at           | 2018-03-23T23:56:35Z               |
| description          |                                     |
| distributed           | False                              |
| external_gateway_info | None                               |
| flavor_id            | None                               |
| ha                    | False                              |
| id                    | 9b5d4516-804a-4c01-9016-3a27fc4197d1 |
| name                  | MyRouter                           |
| project_id            | d5233415ee0b467baec14cbd2d0e1331   |
| revision_number       | None                               |
| routes                |                                     |
| status                | ACTIVE                             |
| tags                  |                                     |
| updated_at           | 2018-03-23T23:56:35Z               |
+-----+-----+
```

Adding a Router Interface

A Router Interface can be added and attached to a subnet with the command:

```
openstack router add subnet <router_name> <subnet_name>
```

In addition, you can specify an *External Gateway* for your router and connect it to the public Network with the following command:

```
openstack router set --external-gateway public <router_name>
```

Deleting Networking Objects

To delete a router with an External Gateway and subnets associated to it, use the following commands:

```
openstack router unset --external-gateway <router_name>
openstack router remove subnet <router_name> <subnet_name>
openstack router delete <subnet>
openstack network delete <network_name>
```

20.3 External Layer2 Connections (Stitching)

Chameleon provides support for sophisticated networking experiments by providing [GENI-style stitching](#). This capability enables users to deploy networking experiments (layer 2 and layer 3) that span Chameleon and other facilities such as [FABRIC](#). Users can create dedicated Chameleon networks directly connected to external facilities and configure custom subnets and routers for handling these external connections. This capability is essential to users interested in experimenting with wide-area networks in a controlled environment or lower-level wide-area protocols (e.g. BGP or other routing protocols) that are not typically configurable by experimenters.

Stitched Chameleon networks are connected to external facilities using one VLAN allocated from a pool of VLANs that are statically provisioned between Chameleon and the external facility. Users can allocate one of these VLANs by making a reservation using Blazar. Currently, externally stitched networks are created by Blazar when the reservation is started. There is a pool of VLANs (3300-3309) between the Chameleon CHI@UC racks to the FABRIC site at StarLight (Chicago). Connections between the TACC site and FABRIC are in development.

The remainder of this document describes how to stitch Chameleon experiments to external resources using FABRIC. In addition to creating the stitched networks, you will need to know how to configure subnets and routers on dynamic VLANs as described in the [Isolated Network VLANs](#) documentation.

Note: Stitching to FABRIC with the Python API is shown in this [Trove Artifact](#)

20.3.1 Configuring a Stitchable Network

Your first step will require creating a stitchable network. Unlike creating other networks on Chameleon, stitchable networks can only be created by first reserving a stitchable VLAN segment. Once you reserve a VLAN segment, your network will be created automatically. To reserve a segment on the appropriate external testbed make sure to include `fabric` as the `stitch_provider` in the `resource_properties` attribute. An example is provided below:

```
openstack reservation lease create --reservation \
resource_type=network,network_name=my-stitchable-network,\
resource_properties='["=$", "$stitch_provider", "fabric"]' \
--start-date "2022-01-01 12:00" --end-date "2022-01-02 12:00" \
my-stitchable-network-lease
```

After your stitched VLAN network is created, you will be able to query for the network to get the specific VLAN that is used by your network. Openstack refers to the VLAN as the `segmentation_id`. The following command will display the VLAN.

```
openstack network show my-stitchable-network --format value -c provider:segmentation_id
```


At this point your Chameleon network is connected to a FABRIC facility port at layer 2. You can now create a FABRIC slice and specify the Chameleon `segmentation_id` to use. This is a layer 2 connection and you can configure higher-level protocols, such as IP, in any way you desire.

20.3.2 Connecting Stitchable Isolated Networks across Chameleon Sites

Coming soon! When FABRIC stitching is available at TACC, you will be able to directly connect your experiments that span UC and TACC Chameleon sites.

20.4 External Layer 3 Connectivity

In addition to configuring networks and floating IPs within a given site (see *Basic Networking*), we provide a shared network named “fabnetv4”, which can route traffic via the FABRIC testbed’s layer3 experimental network. Nodes attached to this network may send traffic to “fabnetv4” on other Chameleon sites, or to anything in the FABNET IPv4 address space, without needing public IPs or traversing a Neutron router.

Besides being easier to set up, this provides lower latency, higher bandwidth, and overall more flexibility for multi-site experiments.

20.4.1 Using FabnetV4

Site	Current status	Fabnet Router IP
UC	Available	10.191.128.1/23
TACC	Available	10.191.130.1/23

At sites where the feature is available, you will be able to see the *fabnetv4* network in Horizon,

▼ Allocated 1

Select networks from those listed below.

	Network	Subnets Associated	Shared	Admin State	Status	
1	fabnetv4	fabnetv4-subnet	Yes	Up	Active	↓
<div><div>ID</div>2a3f6c68-5f98-4d6f-ad9b-f405479aaec2</div> <div><div>Project</div>570aad8999f7499db99eae22fe9b29bb</div> <div><div>External N...</div>No</div> <div><div>Provider Network</div></div> <div><div>Type</div>vlan</div> <div><div>Segmentation ID</div>3499</div> <div><div>Physical Network</div>physnet1</div>						

Or by executing:

```
OS_CLOUD=tacc openstack network show fabnetv4 --fit-width
```

Field	Value

(continues on next page)

(continued from previous page)

admin_state_up	UP	
id	2a3f6c68-5f98-4d6f-ad9b-f405479aaec2	
name	fabnetv4	
project_id	570aad8999f7499db99eae22fe9b29bb	
provider:network_type	vlan	
provider:physical_network	physnet1	
provider:segmentation_id	3499	
router:external	Internal	
shared	True	
status	ACTIVE	
subnets	58283a6f-d834-4ba4-baf3-acfb59dc8648	
+-----+-----+-----+		

You can therefore use it the same way as you would use *sharednet1*, but DHCP will provide an extra route, directing traffic towards *10.128.0.0/10* via the Fabnet router IP for the site.

To launch a server on fabnet:

```
openstack server create \  
--network fabnetv4 \  
..<other usual options>
```

Note: The fabnet router will not send traffic to the public internet. All traffic via floating IPs or otherwise internet bound will still traverse a neutron router at the chameleon site, as with any other isolated network.

After launching your instance, the following traceroutes demonstrate the new paths.

Tracroute from TACC to Google public DNS still traverses the Neutron rotuer:

```
cc@fabnet-v4-test:~$ mtr -n 8.8.8.8 --report  
Start: 2024-03-01T00:55:07+0000  
HOST: fabnet-v4-test  
  1. |-- 10.191.131.254      0.0%   10    0.1   0.1   0.1   0.3   0.0  
  2. |-- 129.114.109.254    0.0%   10    4.5   4.3   1.1  12.8   4.4  
  3. |-- 129.114.0.142     0.0%   10    0.6   6.4   0.5  33.7  12.3  
  4. |-- 192.124.226.21    0.0%   10    6.0   6.1   5.8   6.9   0.3  
  5. |-- 192.124.228.2     0.0%   10    6.3   6.2   6.1   6.3   0.1  
  6. |-- 108.170.231.42    0.0%   10    7.3   7.4   7.2   7.6   0.1  
  7. |-- 142.251.71.113    0.0%   10    6.3   6.3   6.2   6.4   0.1  
  8. |-- 8.8.8.8           0.0%   10    6.3   6.3   6.1   6.4   0.1
```

Traceroute from TACC to closest FABNET router shows a layer 2 path:

```
HOST: fabnet-v4-test  
  1. |-- 10.191.130.1      0.0%   10    0.6   0.7   0.5   0.7   0.1
```

Tracroute from TACC to FABNET router at STAR shows multiple hops through FABNET:

```
cc@fabnet-v4-test:~$ mtr -n 10.191.128.1 -r  
Start: 2024-03-01T01:02:15+0000  
HOST: fabnet-v4-test  
  1. |-- 10.130.158.1      0.0%   10    0.8   0.7   0.7   0.8   0.1
```

(continues on next page)

(continued from previous page)

2. -- 10.130.128.158	0.0%	10	6.6	6.5	6.4	6.6	0.1
3. -- 10.133.128.134	0.0%	10	23.2	23.2	23.1	23.3	0.1
4. -- 10.191.128.1	0.0%	10	42.2	42.2	42.2	42.2	0.0

20.5 Software Defined Networking

Warning: Openflow SDN support is currently not available at the [CHI@UC](#) or [CHI@TACC](#) sites, due to failures in the Corsa switches that supported this feature.

20.5.1 In the meantime

Note: We will be restoring some support once our “Reservable Switch” feature goes live. In the meantime, the FABRIC testbed can be used for networking experiments, and Chameleon supports *External Layer2 Connections (Stitching)* and *External Layer 3 Connectivity* to FABRIC.

20.6 Jumbo Frames

By default, Ethernet frames for networks created on each Chameleon site default to 1500 byte MTU (maximum transmission unit). However, all TOR switches on Chameleon are configured to allow for payloads of up to 9000 bytes. If you would like to experiment with jumbo frames on your private networks or over Layer 2 connections then please follow the steps below to implement.

Note: You will not be able to send jumbo frames out over the public internet, as many commercial networks do not support jumbo frames. You also will not be able to send jumbo frames across two separate tenant networks via an OpenStack router. However, traffic between your nodes or over a stitched layer2 network like ExoGENI can all utilize jumbo frames.

Important: Do not set your MTU value greater than 9000. MTUs greater than 9000 bytes are not supported on Chameleon.

20.6.1 Enabling Jumbo Frames When Creating a Network

Enabling jumbo frames on a new network will ensure that the first Ethernet interface on all newly created baremetal instances will have its MTU set to the value specified.

```
openstack network create --provider-network-type vlan --mtu 9000 \
  --provider-physical-network physnet1 <network_name>
```

Note: You can verify the MTU is correct on your instance with the command `ifconfig eno1`. The first Ethernet interface is typically `eno1` for most Chameleon base images.

20.6.2 Enabling Jumbo Frames on Existing Network

You can also modify the MTU of an existing network using the command below. Please note that this will only effect newly created baremetal instances.

```
openstack network set <network_name> --mtu 9000
```

20.6.3 Enabling Jumbo Frames on Existing Instances

Setting the MTU on your Chameleon network only affects instances on boot to set the first Ethernet interface. If you already have a live baremetal instance then you can simply use the command below on the instance to set MTU manually.

```
sudo ip link set dev eno1 mtu 9000
```

21.1 Introduction

Chameleon provides access to five FPGA nodes. Four nodes are located at [CHI@TACC](#). Each of these nodes is fitted with a [Nallatech 385A board](#) with an Altera Arria 10 1150 GX FPGA (up to 1.5 TFlops), 8 GB DDR3 on-card memory, and dual QSFP 10/40 GbE support. One node is located at [CHI@UC](#). The node is fitted with a [Terasic DE5a-Net board](#) with an [Altera Arria 10 GX 1150 FPGA](#) (up to 1.5 TFlops), 4 GB DDR3 on-card memory, and four QSFP 10/40 GbE support. All FPGA nodes are configured to run OpenCL code, but they can be reconfigured (by a request to our [Help Desk](#)) to run compiled designs prepared with Altera Quartus.

Due to export control limitations, access to the development toolchain requires verification of your user profile. This guide explains how to gain access to the development toolchain and execute code on the FPGA nodes. Briefly, the steps for building an FPGA application are:

- Create a TACC account at the [TACC Portal](#)
- Setup Multi-Factor Authentication for TACC Resources by following [this documentation](#)
- Request access to the FPGA Build Node project at the [Help Desk](#)
- SSH to the `fpga01.tacc.chameleoncloud.org` host to build your FPGA application
- Use `scp` to copy your FPGA application from `fpga01.tacc.chameleoncloud.org` to the FPGA node you wish to run it on

21.2 Development

Chameleon provides a build system that includes the necessary [Altera SDK for OpenCL](#) tools for developing kernels for use on the [Nallatech 385A cards](#) and the [Terasic DE5a-Net card](#), both using the Altera Arria 10 FPGA.

Due to licensing requirements, you must apply for access to the FPGA build system. Submit a ticket through our help system to request access.

FPGA resources are only available at [CHI@TACC](#). Due to TACC's security requirements, multi-factor authentication must be used to access the FPGA build system. You can either use a smartphone app (Apple iOS or Android) or SMS messaging: follow [this documentation](#) to set it up. Once you have set up multi-factor authentication, you can SSH to `fpga01.tacc.chameleoncloud.org` with your Chameleon username and password; you will also be asked for a TACC security token, which will be provided to you via the app or SMS.

Each user's home directory will contain an archive file containing a Hello World OpenCL example: `exm_openc1_hello_world_x64_linux_16.0.tgz`. Extract the archive with the following command:

```
tar -zxvf exm_openc1_hello_world_x64_linux_16.0.tgz
```

Two directories will be extracted: `common` and `hello_world`. Change into the `hello_world` directory.

```
cd hello_world
```

Prior to compiling, load the Quartus environment configuration for either the Nallatech or Terasic board.

Nallatech:

```
module load nallatech
```

Terasic:

```
module load terasic
```

Important:

The host code contains the function `findPlatform(Altera)`, which searches for the “Altera” platform name. It should instead be instructed to search for “Intel(R) FPGA”. This change can be made by editing

```
../hello_world/host/src/main.cpp:  
    findPlatform("Intel(R) FPGA")
```

Compiling an OpenCL kernel often takes a very long time, so it is essential to debug by using the emulation feature of the compiler using `-march=emulator` in the compiler command. Note that the `--board p385a_sch_ax115` parameter is required for the Nallatech board, and the `-board=de5a_net_e1` parameter is required for the Terasic board. These correctly identify the FPGA boards available on Chameleon. Do not alter these parameters or their syntax. In this example, the host application requires the output name to be `hello_world.aocx`, so this parameter must also be unchanged.

Nallatech:

```
aoc --board p385a_sch_ax115 device/hello_world.cl -o bin/hello_world.aocx -march=emulator
```

Terasic:

```
aoc -board=de5a_net_e1 device/hello_world.cl -o bin/hello_world.aocx -march=emulator
```

Build the host application, which is used to execute the OpenCL kernel.

```
make
```

Now run the emulated kernel.

```
env CL_CONTEXT_EMULATOR_DEVICE_INTELFPGA=1 ./bin/host
```

When debugging is complete and the code is ready to be compiled for the FPGA hardware, remove the emulation flag. This may take several hours to complete, so we recommend you run it inside a terminal multiplexer, such as `screen` or `tmux` which are both installed on the build node.

Nallatech:

```
aoc --board p385a_sch_ax115 device/hello_world.cl -o bin/hello_world.aocx
```

Terasic:

```
aoc -board=de5a_net_e1 device/hello_world.cl -o bin/hello_world.aocx
```

21.3 Execution

After completing development of an OpenCL kernel on our build node, the kernel and host application must be transferred and executed on a node with an FPGA accelerator.

When using [CHI@TACC GUI](#) to reserve nodes, use the *Node Type to Reserve* selector and choose *FPGA*. Alternatively, use the [Resource Discovery web interface](#) to reserve a node equipped with an FPGA accelerator card by filtering the node selection using the *with FPGA* button, and clicking *Reserve* at the bottom of the selection. Copy the generated CLI command and use it to create your reservation.

In order to have access to the required runtime environment for using the FPGAs, use the image **CC-CentOS7-FPGA** when launching your instance.

Log in to the instance, download the application code (both `common` and `hello_world` directories) from the build system using `scp`, and change into the `hello_world` directory:

```
scp -r <username>@fpga01.tacc.chameleoncloud.org:~/common .
scp -r <username>@fpga01.tacc.chameleoncloud.org:~/hello_world .
cd hello_world
```

Compile the host application, if necessary.

```
make
```

Program FPGA with the OpenCL kernel, using `acl0` as the device name.

```
aocl program acl0 ./bin/hello_world.aocx
```

Attention: If you are at [CHI@UC](#), please run the following commands (program FPGA as root).

```
sudo -i
source /etc/profile.d/altera.sh
cd /home/cc/hello_world
aocl program acl0 ./bin/hello_world.aocx
```

Execute the host application to run on FPGA.

```
./bin/host
```

You should see an output like the following:

```
Querying platform for info:
=====
CL_PLATFORM_NAME           = Altera SDK for OpenCL
CL_PLATFORM_VENDOR         = Altera Corporation
CL_PLATFORM_VERSION        = OpenCL 1.0 Altera SDK for OpenCL, Version 16.0

Querying device for info:
=====
CL_DEVICE_NAME             = p385a_sch_ax115 : nalla_pcie (aclnalla_pcie0)
CL_DEVICE_VENDOR           = Nallatech ltd
CL_DEVICE_VENDOR_ID        = 4466
CL_DEVICE_VERSION          = OpenCL 1.0 Altera SDK for OpenCL, Version 16.0
```

(continues on next page)

(continued from previous page)

```

CL_DRIVER_VERSION           = 16.0
CL_DEVICE_ADDRESS_BITS     = 64
CL_DEVICE_AVAILABLE       = true
CL_DEVICE_ENDIAN_LITTLE    = true
CL_DEVICE_GLOBAL_MEM_CACHE_SIZE = 32768
CL_DEVICE_GLOBAL_MEM_CACHLINE_SIZE = 0
CL_DEVICE_GLOBAL_MEM_SIZE  = 8589934592
CL_DEVICE_IMAGE_SUPPORT    = true
CL_DEVICE_LOCAL_MEM_SIZE   = 16384
CL_DEVICE_MAX_CLOCK_FREQUENCY = 1000
CL_DEVICE_MAX_COMPUTE_UNITS = 1
CL_DEVICE_MAX_CONSTANT_ARGS = 8
CL_DEVICE_MAX_CONSTANT_BUFFER_SIZE = 2147483648
CL_DEVICE_MAX_WORK_ITEM_DIMENSIONS = 3
CL_DEVICE_MEM_BASE_ADDR_ALIGN = 8192
CL_DEVICE_MIN_DATA_TYPE_ALIGN_SIZE = 1024
CL_DEVICE_PREFERRED_VECTOR_WIDTH_CHAR = 4
CL_DEVICE_PREFERRED_VECTOR_WIDTH_SHORT = 2
CL_DEVICE_PREFERRED_VECTOR_WIDTH_INT = 1
CL_DEVICE_PREFERRED_VECTOR_WIDTH_LONG = 1
CL_DEVICE_PREFERRED_VECTOR_WIDTH_FLOAT = 1
CL_DEVICE_PREFERRED_VECTOR_WIDTH_DOUBLE = 0
Command queue out of order? = false
Command queue profiling enabled? = true
Using AOCX: hello_world.aocx
Reprogramming device with handle 1

Kernel initialization is complete.
Launching the kernel...

Thread #2: Hello from Altera's OpenCL Compiler!

Kernel execution is complete.

```


EXPERIMENT PRECIS

Warning: Precis is no longer supported on the testbed!

22.1 Introduction

Chameleon records experiment setup (OpenStack) events that users performed on the testbed, such as creating leases, creating instances, and setting up networks. Users can request their experiment records from Chameleon using their Chameleon credentials. A report on those experiment records is known as the *Experiment Precis*. An *Experiment Precis* is bounded to a lease. Chameleon defines an *experiment* as a series of testbed setup (OpenStack) events a user performed under a lease of a project. Using *Experiment Precis*, users will be able to analyze, understand and even replay their experiments.

Currently, *Experiment Precis* is provided in JSON format. The following contents are included:

```
{
  "experiment_precis_name": "The name of the experiment precis",
  "experiment_precis_id": "The id of the experiment precis",
  "site": "Which Chameleon site the experiment was performed",
  "testbed_version": "The testbed version at the time the experiment started",
  "report_time_in_utc": "The datetime when the experiment precis was requested (in UTC)",
  "report_time_in_ct": "The datetime when the experiment precis was requested (in CT)",
  "events": "A list of events performed",
  "event_page": "The page number of the events",
  "event_page_size": "The page size of the events",
  "hardware": "Hardwares that were used during the experiment",
  "hardware_page": "The page number of the hardwares",
  "hardware_page_size": "The page size of the hardwares",
  "metric": "Metrics saved in Gnocchi",
  "metric_page": "The page number of the metrics",
  "metric_page_size": "The page size of the metrics"
}
```

22.2 Installation

To request your experiment precis from Chameleon, you need to install the *Chameleon Experiment Precis (cep) Client*. To install `cepclient` on your local machine, run the following command:

```
pip install cepclient
```

To test if `cepclient` is properly installed, run:

```
cep --help
```

22.3 Setting up cep client

Before using the *cep client*, you must configure the authentication-related environment variables via `source the OpenStack RC script` or provide the authentication values as command parameters. If you choose to pass command parameters, use `cep --help` and look for **Authentication Options** for more information.

Note: When using *rc script* for setting up *cep client*, please download and use the **v3** file.

22.4 List experiment precis

You can use `list` command to find all the experiments you have run.

```
cep list
```

The output looks like the following:

```
+-----+-----+-----+-----+
↪      | created_at      | updated_at      | id              | ↪
↪      | name              | lease_id        |                  | ↪
+-----+-----+-----+-----+
↪      | 2018-10-18 09:21:02 | 2018-10-18 09:21:02 | 0fa88391-6b14-465d-a62c-91ad8f6eb920 | ↪
↪      | 0fa88391-6b14-465d-a62c-91ad8f6eb920 | 972b70aa-33ca-42fc-9d4e-e07b2b9df3c3 | ↪
↪      | 2018-10-18 10:05:50 | 2018-10-18 10:05:50 | 93ffbb79-e732-4046-a49d-b223ff8f1bd5 | ↪
↪      | 93ffbb79-e732-4046-a49d-b223ff8f1bd5 | 9f91c7ac-212b-4d46-8f88-1e9db341f41a | ↪
+-----+-----+-----+-----+
```

The *Experiment Precis* will be listed in the reverse order of *creation datetime*, i.e. the latest *Experiment Precis* is listed the first.

For more information, run:

```
cep list --help
```

22.5 Rename experiment precis

Initially, Chameleon sets the name of an *Experiment Precis* the same as its id. However, you can rename it for the convenience of future retrieving. To rename an *Experiment Precis*, run the following command:

```
cep rename --name <new_name> <ep_id or ep_name>
```

Tip: Renaming your experiment precis to a meaningful name will help you 1) mark your *special* experiment; 2) understand what the experiment is about; 3) retrieve your experiment precis.

For more information, run:

```
cep rename --help
```

22.6 Print experiment precis

Finally, you can retrieve all the details about your experiment by using the `print` command.

```
cep print <ep_id or ep_name>
```

The above command will print the requested experiment precis on your terminal in a compact format. To pretty-print the experiment precis, add `--pretty` to the command. To print the experiment precis to a file, add `--output <path/to/file>` to the command.

The following is an example of `cep print` output:

```
{
  "event_page": 0,
  "event_page_size": -1,
  "events": [
    {
      "event_time": "2018-10-18 15:05:50",
      "event_type": "lease.create",
      "metadata": {
        "end_date": "2018-10-19T15:05:00.000000",
        "start_date": "2018-10-18T15:06:00.000000"
      },
      "resource_id": "9f91c7ac-212b-4d46-8f88-1e9db341f41a",
      "service": "blazar"
    },
    {
      "event_time": "2018-10-18 15:06:05",
      "event_type": "lease.event.start_lease",
      "metadata": {
        "end_date": "2018-10-19T15:05:00.000000",
        "start_date": "2018-10-18T15:06:00.000000"
      },
      "resource_id": "9f91c7ac-212b-4d46-8f88-1e9db341f41a",
      "service": "blazar"
    }
  ],
}
```

(continues on next page)

(continued from previous page)

```

...

{
  "event_time": "2018-10-19 15:05:11",
  "event_type": "lease.event.end_lease",
  "metadata": {
    "end_date": "2018-10-19T15:05:00.000000",
    "start_date": "2018-10-18T15:06:00.000000"
  },
  "resource_id": "9f91c7ac-212b-4d46-8f88-1e9db341f41a",
  "service": "blazar"
}
],
"experiment_precis_id": "93ffbb79-e732-4046-a49d-b223ff8f1bd5",
"experiment_precis_name": "zhenz-test-2",
"hardware": [
  {
    "architecture": {
      "platform_type": "x86_64",
      "smp_size": 2,
      "smt_size": 48
    },
    "bios": {
      "release_date": "03/09/2015",
      "vendor": "Dell Inc.",
      "version": 1.2
    },
    "chassis": {
      "manufacturer": "Dell Inc.",
      "name": "PowerEdge R630",
      "serial": "8Q28C42"
    },
    "gpu": {
      "gpu": false
    },
    "links": [
      {
        "href": "/sites/uc/clusters/chameleon/nodes/b0525159-5c95-4b71-83f2-
↪b8d6bdd2acd2",
        "rel": "self",
        "type": "application/vnd.grid5000.item+json"
      },
      {
        "href": "/sites/uc/clusters/chameleon",
        "rel": "parent",
        "type": "application/vnd.grid5000.item+json"
      },
      {
        "href": "/sites/uc/clusters/chameleon/nodes/b0525159-5c95-4b71-83f2-
↪b8d6bdd2acd2/versions/53c90ef0512d5013ee30d431cd62e68bfd34d4ca",
        "rel": "version",

```

(continues on next page)

(continued from previous page)

```

        "type": "application/vnd.grid5000.item+json"
    },
    {
        "href": "/sites/uc/clusters/chameleon/nodes/b0525159-5c95-4b71-83f2-
↪b8d6bdd2acd2/versions",
        "rel": "versions",
        "type": "application/vnd.grid5000.collection+json"
    }
],
"main_memory": {
    "humanized_ram_size": "128 GiB",
    "ram_size": 134956859392
},
"monitoring": {
    "wattmeter": false
},
"network_adapters": [
    {
        "bridged": false,
        "device": "eno1",
        "driver": "bnx2x",
        "interface": "Ethernet",
        "mac": "44:a8:42:15:c4:dd",
        "management": false,
        "model": "NetXtreme II BCM57800 1/10 Gigabit Ethernet",
        "mounted": true,
        "rate": 100000000000,
        "vendor": "Broadcom Corporation"
    },
    {
        "bridged": false,
        "device": "eno2",
        "driver": "bnx2x",
        "interface": "Ethernet",
        "mac": "44:a8:42:15:c4:df",
        "management": false,
        "model": "NetXtreme II BCM57800 1/10 Gigabit Ethernet",
        "mounted": false,
        "rate": 100000000000,
        "vendor": "Broadcom Corporation"
    },
    {
        "bridged": false,
        "device": "eno3",
        "driver": "bnx2x",
        "interface": "Ethernet",
        "mac": "44:a8:42:15:c4:e1",
        "management": false,
        "model": "NetXtreme II BCM57800 1/10 Gigabit Ethernet",
        "mounted": false,
        "rate": 100000000000,
        "vendor": "Broadcom Corporation"
    }
]

```

(continues on next page)

(continued from previous page)

```

    },
    {
      "bridged": false,
      "device": "eno4",
      "driver": "bnx2x",
      "interface": "Ethernet",
      "mac": "44:a8:42:15:c4:e3",
      "management": false,
      "model": "NetXtreme II BCM57800 1/10 Gigabit Ethernet",
      "mounted": false,
      "rate": 10000000000,
      "vendor": "Broadcom Corporation"
    }
  ],
  "node_type": "compute_skylake",
  "placement": {
    "node": 14,
    "rack": 1
  },
  "processor": {
    "cache_l1": null,
    "cache_l1d": 32768,
    "cache_l1i": 32768,
    "cache_l2": 262144,
    "cache_l3": 31457280,
    "clock_speed": 31000000000,
    "instruction_set": "x86-64",
    "model": "Intel Xeon",
    "other_description": "Intel(R) Xeon(R) CPU E5-2670 v3 @ 2.30GHz",
    "vendor": "Intel",
    "version": "E5-2670 v3"
  },
  "storage_devices": [
    {
      "device": "sda",
      "driver": "megaraid_sas",
      "humanized_size": "250 GB",
      "interface": "SATA",
      "model": "ST9250610NS",
      "rev": "AA63",
      "size": 250059350016,
      "vendor": "Seagate"
    }
  ],
  "supported_job_types": {
    "besteffort": false,
    "deploy": true,
    "virtual": "ivt"
  },
  "type": "node",
  "uid": "b0525159-5c95-4b71-83f2-b8d6bdd2acd2",
  "version": "53c90ef0512d5013ee30d431cd62e68bfd34d4ca"

```

(continues on next page)

(continued from previous page)

```

    }
  ],
  "hardware_page": 0,
  "hardware_page_size": 25,
  "metric_page": 0,
  "metric_page_size": 25,
  "metrics": [
    {
      "instance_id": "44ad06ee-41d7-48f9-a52a-179030754707",
      "metric_id": "dd22e02386714516a913d966659617eb",
      "metric_name": "interface-eno1@if_dropped"
    },
    {
      "instance_id": "44ad06ee-41d7-48f9-a52a-179030754707",
      "metric_id": "512ac94754b64906a12960d1f0a929c9",
      "metric_name": "interface-eno1@if_errors"
    },
    ...
    {
      "instance_id": "44ad06ee-41d7-48f9-a52a-179030754707",
      "metric_id": "89cec271c0fb477b9e7bb37ad3df1331",
      "metric_name": "memory@memory.slab_recl"
    }
  ],
  "report_time_in_ct": "2018-10-19 11:06:51",
  "report_time_in_utc": "2018-10-19 16:06:51",
  "site": "CHI_DEV_UC",
  "testbed_version": "702d4d47ab21c890c0bb146f4e0256f618264487"
}

```

The `events` section is a list of testbed events ordered by event timestamp. The `hardware` section contains information of all the nodes that were used in the experiment. The hardware information is retrieved by using the same method as *the Resource Discovery*. The `metrics` section is a list of the metrics captured during the experiment. The *Experiment Precis* only contains the `instance_id`, `metric_id`, and `metric_name` in the `metrics` list. You can use *the openstack metric command line* to get all the measurements of a particular metric over time for an instance.

For more information, run:

```
cep print --help
```

Important: Chameleon only keeps an experiment precis for **180 days**. Please make sure to save your experiment precis you'd like to keep for a longer time by using `cep print` command. You can output it to a file and keep it as a record.

22.6.1 Pagination

In the case of “large” experiment with large number of nodes and metrics, events, hardware, and metrics are printed in pages. By default, the page number is set to 0 and the page size is set to 25. However, you can tune the pagination by specifying the following parameters:

```
--event-page-size EVENT_PAGE_SIZE
    Page size for event; ignored if event is excluded; set
    to negative value to show all
--event-page EVENT_PAGE
    Page number for event; ignored if event is excluded
--metric-page-size METRIC_PAGE_SIZE
    Page size for metric; ignored if metric is excluded;
    set to negative value to show all
--metric-page METRIC_PAGE
    Page number for metric; ignored if metric is excluded
--hardware-page-size HARDWARE_PAGE_SIZE
    Page size for hardware; ignored if hardware is
    excluded; set to negative value to show all
--hardware-page HARDWARE_PAGE
    Page number for hardware; ignored if hardware is
    excluded
```

Tip: To show all, set page size to a negative value. If page size is negative, page parameter will be ignored. Negative value for page is not allowed.

22.6.2 Filters

The cep tool provides multiple filters to help you focus on the contents you care.

Event Filters

- To exclude all the events from the *Experiment Precis*, use `--exclude-event`.
- To include or exclude services, use `--include-services` and/or `--exclude-services`. For example, if you only want to print blazar (reservation) and nova (instance) related events, run the following command:

```
cep print --pretty --include-services blazar,nova <ep_id or ep_name>
```

- You can exclude event metadata by passing `--exclude-event-metadata`.
- You can apply datetime filters to your events. For example, to print events up to 2018-10-05 00:00:00, run:

```
cep print --pretty --end-datetime "2018-10-05 00:00:00" <ep_id or ep_name>
```

Or to print events from 2018-10-05 09:00:00 to 2018-10-05 17:00:00, run:

```
cep print --pretty -start-datetime "2018-10-05 09:00:00" --end-datetime "2018-10-05↵
↵17:00:00" <ep_id or ep_name>
```

Note: When using datetime filters, use datetime in UTC.

Metric Filters

- To exclude metrics from the *Experiment Precis*, use `--exclude-metric`.

Hardware Filters

- To exclude hardware information from the *Experiment Precis*, use `--exclude-hardware`.

TROVI SHARING PORTAL

Chameleon Trovi is a sharing portal that allows you to share digital research and education artifacts, such as packaged experiments, workshop tutorials, or class materials. Each research artifact is represented as a deposition (a remotely accessible folder) where a user can put Jupyter notebooks, links to images, orchestration templates, data, software, and other digital representations that together represent a focused contribution that can be run on Chameleon. Users can use these artifacts to recreate and rerun experiments or class exercises on a Jupyter Notebook within Chameleon. They can also create their own artifacts and publish them directly to Trovi from within *Chameleon's Jupyter server*.

To get started, find the “Trovi” dropdown option under the “Experiment” section of chameleoncloud.org. Once you're on the Trovi homepage, you'll see a list of publicly available experiments and other digital artifacts. You can now browse those artifacts or upload your own.

23.1 Browsing artifacts

Trovi allows you to browse artifacts, presented in a scrolling list format. On the right hand side, there are multiple filtering options. The “All” choice shows you all of the artifacts you have access to. You can also see how many times people have downloaded and launched your notebook with the icons in the bottom left corner of an artifact.

Note: Some Trovi artifacts are supported by the Chameleon team and are denoted with a small Chameleon logo. You can contact the [Help Desk](#) if you are using these artifacts and encounter issues.

23.1.1 Launching an artifact

The most powerful feature available via Trovi is the ability to re-launch the available artifacts within Chameleon. Clicking “Launch with JupyterHub” will open a new Jupyter Notebook server with the artifacts downloaded (we support artifacts up to 500MB in total size, please contact the [Help Desk](#) if you need more space). The animation below shows how easy it is:



Fig. 1: The “Trove” option under the “Experiment” section takes you to Trove.

Fig. 2: Clicking the “Launch with JupyterHub” button to import a Trove artifact into your own Jupyter server.

23.2 Packaging shared artifacts

You can publish new artifacts to Trovi either from your primary Jupyter server or by editing a previously-shared artifact. In the latter case, you are effectively creating a new “forked” artifact owned by you.

When you’ve finished creating or making changes to an experiment, in the Jupyter interface, select the directory (not a single file) you wish to package. Then, click on the “Share” tab and select “Package as a new artifact”. Your artifact is now packaged and uploaded to Chameleon file storage, and you’ll be prompted to fill out descriptions about the artifact. Don’t worry if you want to change this later—you will be able to *edit them on the Trovi portal or within Jupyter*.

Congratulations! Your artifact is now uploaded to Trovi—but to make it accessible to others you need to *adjust its sharing settings*.

23.2.1 Saving new versions

If you make changes to your artifact, you can submit an updated version. Within Jupyter, you navigate to the “Sharing” tab, but this time you click “Create new artifact version”. The different versions are viewable on the Trovi portal after clicking on the artifact.

23.2.2 Editing artifacts

You can edit an artifact’s metadata, including its title, description, and list of authors at any time via the Jupyter interface. To delete single artifact versions, click the “trashcan” icon next to it in the edit view. To delete the entire artifact, click the red “Delete All” button.

Note: Any artifacts published to *Zenodo* cannot be deleted.

This edit view is also available from Trovi via the “Edit” button.

Creating a version from Git

Under an artifact’s edit settings, you will also see a button for creating a new version from Git. This will allow you to enter a Git URL, the same one used to clone your repository, and also a reference (lease as HEAD for the latest commit). When a user launches your artifact, their notebook will checkout the specified commit.

23.2.3 Adjusting sharing settings

When you first upload your packaged artifact to Trovi, its visibility is set as private, meaning only you can see or launch it. There are multiple options to change the visibility of the artifact, and you have the option to decide how visible you want it to be.

1. **Publish with DOI:** this option allows you to *publish a version of your artifact to Zenodo* and receive a DOI, which you can use to cite your artifact in, e.g., an academic paper.
2. **Publish without DOI:** this option allows any Chameleon user to find and launch your artifact. It can be useful if you want to distribute the artifact widely but do not necessarily wish to publish it to Zenodo and get a DOI for citation.

3. **Share via private link:** this option allows you to share the experiment to select people, like individual colleagues, advisors, or students. Anybody in possession of the link can view and launch any version of the artifact.

To make your artifact shareable, select it in Trovi, click “Share”, and check the box before “Enable all users to find and share”.

Assigning Roles to Other Users

<input type="text" value="admin@uchicago.edu"/>	<input type="checkbox"/> collaborator <input checked="" type="checkbox"/> administrator
<input type="text" value="foo@bar.baz"/>	<input type="checkbox"/> collaborator <input checked="" type="checkbox"/> administrator
<input type="text" value="Email"/>	<input type="checkbox"/> collaborator <input type="checkbox"/> administrator
<input type="text" value="Email"/>	<input type="checkbox"/> collaborator <input type="checkbox"/> administrator

You can assign roles to other users which allow them to collaborate on your artifacts. There are two roles: **Collaborator** and **Administrator**.

Collaborators are allowed to edit artifact metadata, upload new versions, delete old versions, and share private artifacts.

Administrators have full control over the artifact, including assigning roles to other users.

Artifact owners cannot have their Administrator privileges removed.

Publishing to Zenodo

Attention: You can only request a DOI for artifacts uploaded via the Jupyter interface. You cannot request a DOI for an artifact version uploaded via git.

Trovi is intended for sharing work in progress with a limited group of “friends and family”. However, once you complete your experiment package you may want to publish it so that you can reference it from your paper. To do that Chameleon supports integration with Zenodo, an open-access storage repository backed by CERN, for permanent artifact hosting. To share your artifact and store it on Zenodo, go to the “Share” page for the artifact. On the right-hand side you’ll see a list of all versions you’ve saved. Pick the version you want to publish to Zenodo and check “Request DOI”, then click “Save.”

Important: Once published, **Zenodo artifacts cannot be deleted** and are additionally **publicly available**. Your artifact will appear in Trovi in the “Public” section, and any Chameleon user can access it, as can anybody on the Internet via Zenodo’s own listing.

If you wish to make your artifact public but don’t to publish it, use the “Publish without DOI” option. With this option it is possible to make the artifact private later on if you wish; this is not possible when publishing to Zenodo.

You can only request a DOI one time per artifact. If you want to update your experiment files and request a second DOI, you should instead create a new artifact.

This also creates a DOI, which you can easily include in your paper. The artifacts shared on Zenodo also appear on Trovi.

23.2.4 Importing an artifact

Instead of creating an artifact inside Jupyterhub, you can package an existing Git repository into an artifact. When a user launches the artifact, the contents of the repository will be added to a Jupyter notebook.

To create an artifact, click “Import Artifact” on the sidebar of Trovi. You are first asked for the artifact’s metadata. At the bottom of the form, there is a button for “Import from Git.” After clicking this, you will need to enter a git remote URL, and choose which commit to tie the version to.

To update the artifact, you must create a *new version*. This ensures that a given version of your artifact always has the same contents.

23.2.5 Exporting via git

If you wish to move your code and notebooks outside of your Jupyter notebook, one option is to export it into a git repository.

1. Click the “+” button on the top left of your notebook, and choose “Terminal”.
2. Run the command `cd work`. If there is a specific directory you wish to export, you can `cd` again into it.
3. Follow the instructions to set up a repository per your git host. For GitHub see [this document](#).
4. After the repository is setup, you should be able to commit and push with the git CLI.

DAYPASS

Normally, only Chameleon users with active allocations are able to launch and view Trovi artifacts. To allow anyone to launch an artifact, we also provide daypass. This allows for a non-Chameleon user to have access to Chameleon for a limited amount of time, using a small, separate allocation. People interested in reproducing your project will send requests to the managers of a project. If approved, the requesting user will receive an email invitation to join a reproducibility project. When they accept, they can use this project to run your artifact. After the specified time limit, they will be automatically removed from this project. Daypass can be enabled on an artifact-by-artifact basis in Trovi.

24.1 Allowing Reproducibility Requests

First, the owner of an artifact must permit reproducibility requests. This can be revoked at any time, preventing future requests. Additionally, you must also give your artifact a value for “Hours a user has to reproduce.” This value specifies how long a user will have access to Chameleon for. Consider how long it takes to run your experiment from start to finish as a lower bound for this value. The artifact owner must also assign their artifact to a project via the dropdown selector. As these requests are granting access to Chameleon resources, this is needed to tie granted requests to a PI.

These fields can be accessed by navigating to an artifact’s detail page, and then selecting “Share.” At the bottom of the share page, you will see the below forms, which are the project assignment, the enabling of reproducibility requests, and the hours to reproduce.

A person without an active Chameleon allocation is unable to launch and reproduce your artifact. If you enable reproducibility requests, users can send requests, and if granted, they will be given a small temporary allocation on a separate project. While usage by reproducing users will not count towards your project's main allocation, the PI is nonetheless accountable for their responsible usage of Chameleon resources. All reproducibility requests therefore require PI approval.

☐ Enable reproducibility requests

Hours a user has to reproduce

Hours a user has to reproduce

After these items are saved, a reproducibility allocation request is automatically made under your PI’s name. Your artifact should now appear with a “Request Daypass” button below the “Launch” button. The “Launch” button will not

appear for users that are not a member of an active Chameleon project.

24.1.1 Requesting a Daypass

When another researcher wishes to reproduce your artifact (or when you wish to do so for another’s artifact), selecting “Request Daypass” will display a form asking for a name, institution, and the reason for reproducing the artifact. For artifacts under your project/allocation, this gives you oversight and discretion as to who you grant access to, as ultimately you’re responsible for usage under your allocation (including reproducibility allocations). For artifacts you wish to explore, it gives you the chance to reach out to the project owners and explain why you are interested in their work.

The screenshot shows the Chameleon Cloud web interface. At the top is a green navigation bar with the Chameleon logo and links for 'About', 'Learn', 'Experiment', and 'Blog'. A 'Log in' button is in the top right. Below the navigation bar is a breadcrumb trail: 'Artifacts / Getting Started with Chameleon: Power management experiment'. The main content area is divided into two columns. The left column contains the artifact details: the title 'Getting Started with Chameleon: Power management experiment', a description 'This notebook is a short example of how to use Chameleon notebooks to run a simple experiment, and analyze the data, using the python-chi interface.', 'Estimated duration: 1 hour', 'Support contact: help@chameleoncloud.org', a version indicator '2' with a notebook icon and timestamp 'Sept. 29, 2021, 12:44 p.m.', and a list of authors 'Jason Anderson (University of Chicago)' and 'Mark Powers (University of Chicago)'. The right column features a green 'Request day pass' button, a paragraph explaining that users without an active allocation can request a temporary one, and a 'Versions' table.

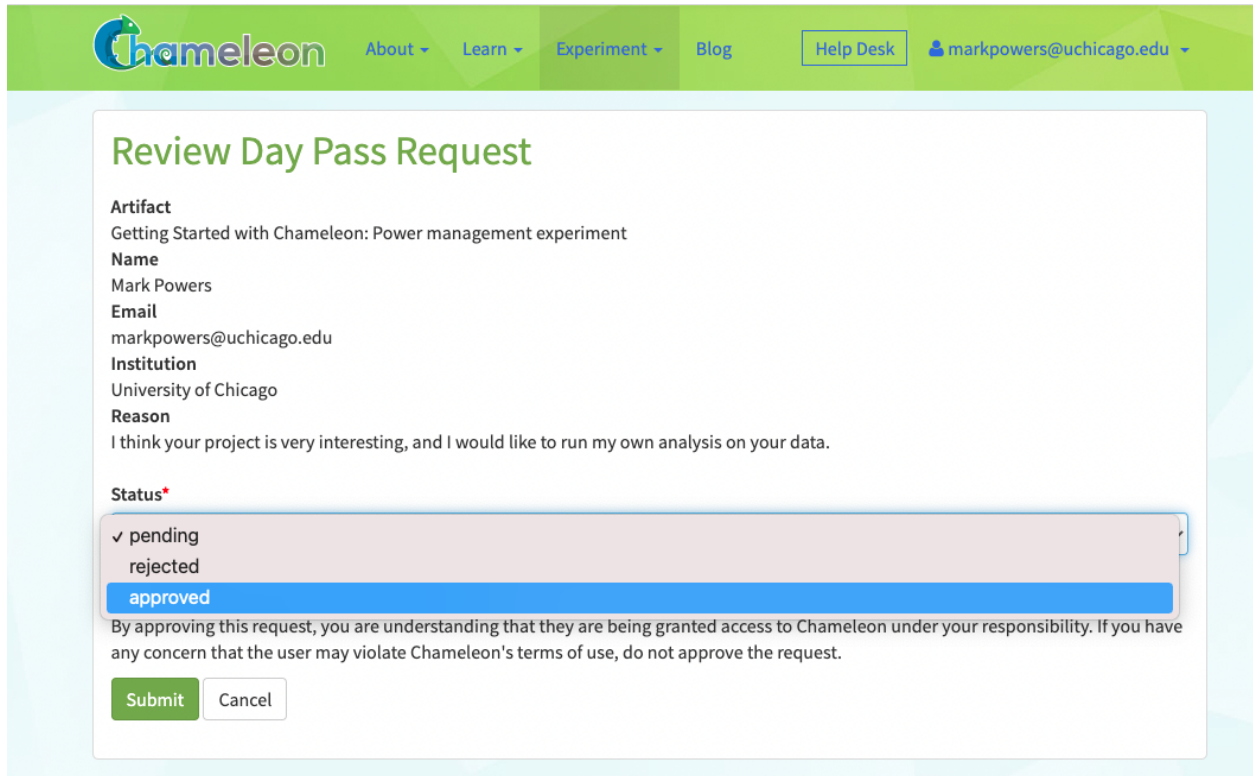
Versions	
Version 2	Sept. 29, 2021, 12:43 p.m.
Version 1	Sept. 29, 2021, 12:37 p.m.

After submitting the form, the managers (and PI) of the project associated with the artifact will receive an email informing them of the request.

24.1.2 Reviewing a Daypass Request

After receiving an email with the daypass request, PIs and project managers can navigate to the review page by clicking the link in the email. Here, they will see all of the details submitted with the request. A decision can be made by choosing “approved” or “rejected” in the selector, and then clicking submit.

After this decision is made, an email is sent to the requestor with the result. If the request is approved, an invitation is sent to the user.



The screenshot shows the Chameleon Cloud interface. The top navigation bar is green with the Chameleon logo and links for About, Learn, Experiment, Blog, Help Desk, and a user profile for markpowers@uchicago.edu. The main content area is titled 'Review Day Pass Request'. It contains the following information:

- Artifact:** Getting Started with Chameleon: Power management experiment
- Name:** Mark Powers
- Email:** markpowers@uchicago.edu
- Institution:** University of Chicago
- Reason:** I think your project is very interesting, and I would like to run my own analysis on your data.

The **Status*** dropdown menu is open, showing three options: 'pending' (with a checkmark), 'rejected', and 'approved' (highlighted in blue). Below the dropdown, a warning message states: 'By approving this request, you are understanding that they are being granted access to Chameleon under your responsibility. If you have any concern that the user may violate Chameleon's terms of use, do not approve the request.' At the bottom are 'Submit' and 'Cancel' buttons.

24.1.3 Using an Invitation

If your daypass request is approved, an email will be sent to you with an invite link. After clicking this link, you will be automatically added to the project. The email will also mention how long the invitation is for. When the invite is accepted, you will be taken to the project page for the reproducibility project. Please note the ID of the project (CHI-XXXXXX), which may be needed to configure an artifact.

Next, you can navigate back to the original artifact URL you were given. The “Launch” button can be used now to start running the artifact.

After the duration for the invite has passed, you will be automatically removed from the project.

25.1 Introduction

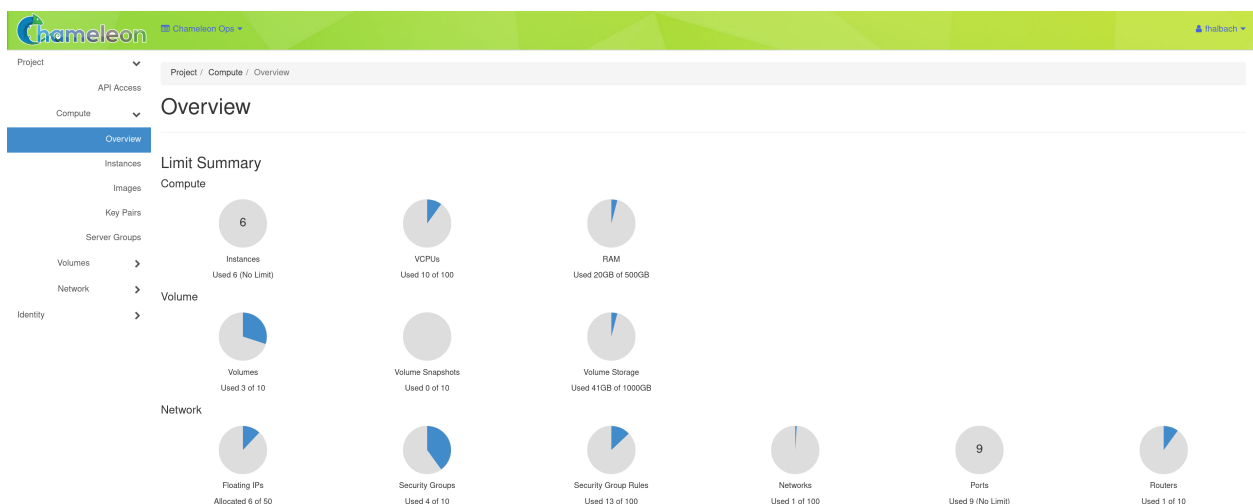
OpenStack is an Infrastructure as a Service (IaaS) platform that allows you to create and manage virtual environments. Chameleon provides an installation of OpenStack *Xena* using the KVM virtualization technology at the [KVM@TACC](#) site. Since the KVM hypervisor is used on this cloud, any virtual machines you upload must be compatible with KVM.

This documentation provide basic information about how to use the OpenStack web interface and provides some information specific to using OpenStack KVM on Chameleon. The interface is similar to the bare metal sites [CHI@TACC](#) and [CHI@UC](#). However, the resources that you are using are virtual, rather than being tied to physical nodes. Familiarity with some concepts, such as *Key Pairs* are also required for KVM.

25.2 Work with KVM using the GUI

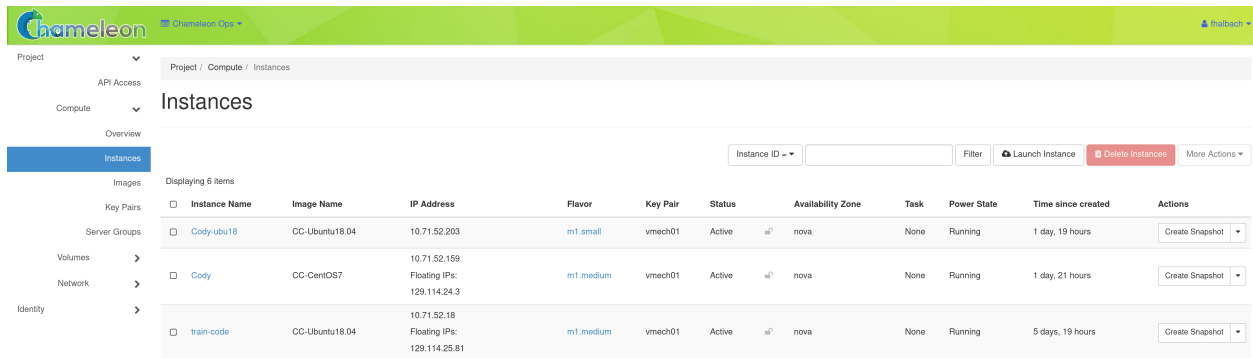
An easy way to use OpenStack KVM on Chameleon is via the GUI, which is similar to the GUIs for [CHI@TACC](#) and [CHI@UC](#). You log into the web interface using your Chameleon username and password.

After a successful log in, you will see the *Overview* page as shown below. This page provides a summary of your current and recent usage and provides links to various other pages. Most of the tasks you will perform are done via the menu on the lower left and will be described below. One thing to note is that on the left, your current project is displayed. If you have multiple Chameleon projects, you can change which of them is your current project. All of the information displayed and actions that you take apply to your current project. So in the screen shot below, the quota and usage apply to the current project you have selected and no information about your other projects is shown.



25.2.1 Managing Virtual Machine Instances

One of the main activities you'll be performing in the GUI is management of virtual machines, or instances. Go to *Project > Compute > Instances* in the navigation sidebar. For instances that you have running, you can click on the name of the instance to get more information about it and to access the VNC interface to the console. The dropdown menu to the right of the instance lets you perform a variety of tasks such as suspending, terminating, or rebooting the instance.



25.2.2 Launching Instances

To launch an *Instance*, click the *Launch Instance* button. This will open the *Launch Instance* dialog.

Launch Instance

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

nova

Count *

1

Total Instances (No Limit)

6 Current Usage

1 Added

Cancel Back Next Launch Instance

On the *Details* tab, provide a name for this instance (to help you identify instances that you are running).

Next, go to the *Source* tab to select media to launch.

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				

▼ Available 118

Select one

Q

Click here for filters.

×

Name	Updated	Size	Type	Visibility
ark-cenOS	10/17/19 11:21 AM	792.00 MB	iso	Public
CC-C7-autologin	10/17/19 11:08 AM	8.00 GB	raw	Public
CC-CentOS7	2/10/20 1:46 PM	4.04 GB	raw	Public
CC-CentOS7-1602.0	1/22/20 11:48 AM	2.95 GB	raw	Public

Select the *Boot Source* of the instance, which is either an *Image*, an *Instance Snapshot* (an image created from a running virtual machine), a *Volume* (a persistent virtual disk that can be attached to a virtual machine), or a “Volume Snapshot”. If you select “Image” as the *Boot Source*, the *Image Name* dropdown presents a list of virtual machine images that we have provided, that other Chameleon users have uploaded and made public, or images that you have uploaded for yourself. If you select *Boot from snapshot*, the *Instance Snapshot* dropdown presents a list of virtual machine images that you have created from your running virtual machines.

Go to the *Flavor* Tab and select the amount of resources (Flavor) to allocate to the instance.

Flavors refer to the virtual machine’s assigned memory and and disk size. Different images and snapshots may require a larger Flavor. For example, the CC-CentOS7 image requires at least an `m1.small` flavor.

Tip: If you select different flavors from the Flavor dropdown, their characteristics are displayed on the right.

When you are finished with this step, go to the *Key Pair* Tab.

Select an SSH keypair that will be inserted into your virtual machine. You will need to select a keypair here to be able to access an instance created from one of the public images Chameleon provides. These images are not configured with a default root password and you will not be able to log in to them without configuring an SSH key.

Then, go to the *Security Groups* Tab.

If you have previously defined *Security Groups*, you may select them here. Alternatively, you can configure them later.

Set up network using *Network* tab.

Launch Instance

Details *

Source *

Flavor *

Networks *

Network Ports

Security Groups

Key Pair

Configuration

Server Groups

Scheduler Hints

Metadata

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

Allocated

Name	VCPUS	RAM	Total Disk	Root Disk	Ephemeral Disk	Public
Select an item from Available items below						

Available 5

Select one

Click here for filters.

Name	VCPUS	RAM	Total Disk	Root Disk	Ephemeral Disk	Public
m1.tiny	1	512 MB	1 GB	1 GB	0 GB	Yes
m1.small	1	2 GB	20 GB	20 GB	0 GB	Yes
m1.medium	2	4 GB	40 GB	40 GB	0 GB	Yes
m1.large	4	8 GB	40 GB	40 GB	0 GB	Yes
m1.xlarge	8	16 GB	40 GB	40 GB	0 GB	Yes

Cancel

Back

Next >

Launch Instance

Select which network should be associated with the instance. Click the Up arrow next to your project's private network (PROJECT_NAME-net), not ext-net.

Now you can launch your instance by clicking on the *Launch* button and the *Instances* page will show progress as it starts.

25.2.3 Associating a Floating IP Address

You may assign a Floating IP Address to your Instance by selecting *Associate Floating IP* in the dropdown menu next to your Instance on the *Instances* page.

This process is similar to *Associate a Floating IP* on CHI@TACC and CHI@UC bare metal sites.

Launch Instance

Details *

Source *

Flavor *

Networks *

Network Ports

Security Groups

Key Pair

Configuration

Server Groups

Scheduler Hints

Metadata

A key pair allows you to SSH into your newly created instance. You may select an existing key pair, import a key pair, or generate a new key pair.

+ Create Key Pair

Import Key Pair

Allocated

Displaying 1 item

Name	Fingerprint
> fhlib_key2	c2:e5:d4:3f:2f:72:15:e7:d2:cf:b2:6e:3e:82:73:4a

Displaying 1 item

▼ Available 0

Select one

Click here for filters.

Displaying 0 items

Name	Fingerprint
No items to display.	

Displaying 0 items

Cancel

< Back

Next >

Launch Instance

Launch Instance

Details *

Source *

Flavor *

Networks *

Network Ports

Security Groups

Key Pair

Configuration

Server Groups

Scheduler Hints

Metadata

Select the security groups to launch the instance in.

▼ Allocated 1

Name	Description
> default	Default security group

▼ Available 3

Select one or more

Click here for filters.

Name	Description
> vscode	
> Allow ICMP	
> Allow SSH	

Cancel

< Back

Next >

Launch Instance

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

Q

Click here for filters.

×

Network	Subnets Associated	Shared	Admin State	Status
> Chameleon Ops-net	Chameleon Ops-subnet	No	Up	Active
> sharednet1	sharednet1-subnet	Yes	Up	Active

✕ Cancel

< Back

Next >

Launch Instance

Instances

Instance ID ▾

Filter

Launch Instance

Delete Instances

More Actions ▾

Displaying 6 items

<input type="checkbox"/>	Instance Name	Image Name	IP Address	Flavor	Key Pair	Status	Availability Zone	Task	Power State	Time since created	Actions
<input type="checkbox"/>	Cody-ubu18	CC-Ubuntu18.04	10.71.52.203	m1.small	vmec01	Active	nova	None	Running	1 day, 20 hours	Create Snapshot
<input type="checkbox"/>	Cody	CC-CentOS7	10.71.52.159	m1.medium	vmec01	Active	nova	None	Running	1 day, 22 hours	Associate Floating IP Attach Interface Detach Interface

25.2.4 Key Pairs

You will need to import or create SSH *Key Pairs*. This process is similar to the process performed on [CHI@TACC](#) and [CHI@UC](#) bare metal sites.

25.2.5 Security Groups

Security Groups allow you to specify what inbound and outbound traffic is allowed or blocked to Instances. Unlike the [CHI@TACC](#) and [CHI@UC](#) bare metal sites, [KVM@TACC](#) observes Security Groups for Instances.

Note: By default, all inbound traffic is blocked to [KVM@TACC](#) Instances, including SSH. You must apply a Security Group that allows TCP port 22 inbound to access your instance via SSH.

To create a Security Group, click *Projects > Network > Security Groups* in the navigation side bar.

The screenshot shows the Chameleon Cloud web interface. The left sidebar has a navigation menu with categories like Project, API Access, Compute, Volumes, Network, and Identity. Under the 'Network' category, 'Security Groups' is selected. The main panel shows the 'Security Groups' page with a breadcrumb 'Project / Network / Security Groups'. There is a search bar and buttons for '+ Create Security Group' and '- Delete Security Groups'. A table lists four security groups:

Name	Security Group ID	Description	Actions
Allow ICMP	4db27c8-d5bc-41f1-b153-4218a081899d		Manage Rules
Allow SSH	92c3ac13-4c0b-425b-9685-ed077717b996		Manage Rules
default	e92226e1-1883-4320-b63b-24ec01b25acb	Default security group	Manage Rules
vscode	08e93259-6e77-4404-ba39-523873cac038		Manage Rules

Click the **+Create Security Group** button to open the *Create Security Group* page.

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.

Cancel

Create Security Group

Enter a *Name* for your *Security Group*, and optionally provide a *Description*. Then click the *Create Security Group* button. Now, you should see your *Security Group* listed on the *Access and Security* page.

25.2. Work with KVM using the GUI

211

Project / Network / Security Groups

Security Groups

Displaying 5 items

Filter [+ Create Security Group](#) [Delete Security Groups](#)

<input type="checkbox"/>	Name	Security Group ID	Description	Actions
<input type="checkbox"/>	Allow ICMP	4dfb27c8-d5bc-41ff-b1b3-4218a081899d		Manage Rules
<input type="checkbox"/>	Allow SSH	92c3acd3-4c0b-425b-9685-ed077717fb96		Manage Rules
<input type="checkbox"/>	default	e92226e1-f883-4320-b63b-24ec01b25acb	Default security group	Manage Rules
<input type="checkbox"/>	newgroup	09e352a5-c821-4b44-ba22-e3af0001ce45		Manage Rules
<input type="checkbox"/>	vscode	08e93259-6e77-4404-ba39-523873cac038		Manage Rules

Displaying 5 items

Click the *Manage Rules* button in the *Action* column to open the *Manage Security Group Rules* page.

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

Manage Security Group Rules: newgroup (09e352a5-c821-4b44-ba22-e3af0001ce45)

Displaying 2 items

[+ Add Rule](#) [Delete Rules](#)

<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	-	-	Delete Rule
<input type="checkbox"/>	Egress	IPv6	Any	Any	:::0	-	-	Delete Rule

Displaying 2 items

The default Security Group allows outbound IPv4 and IPv6 traffic for *Any IP Protocol* and *Port Range*. If no entry for *Ingress*, no inbound traffic will be allowed. You may add an additional rule by clicking on the *+Add Rule* to open the *Add Rule* dialog.

In this dialog, you can specify *Custom TCP Rule* (or *Custom UDP Rule* or *Custom ICMP Rule*), a *Direction* (*Ingress* for inbound traffic to your Instance or *Egress* for outbound traffic) and a *Port*. Alternatively, you can use a pre-defined rule in the *Rule* dropdown, such as *SSH*. when you are finished, click *Add*.

25.2.6 Adding a Security Group to an Instance

Once you have defined a *Security Group*, you may apply it to an Instance by clicking *Project > Compute > Instances* in the navigation sidebar and clicking the *Edit Security Groups* option in the *Actions* dropdown.

The *Security Groups* tab in the *Edit Instance* dialog will pop up.

You may click the *+* button next to the Security Group you wish to apply in the *All Security Groups* list on the left. Once you are finished, click *Save* to finish the process.

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

Instances

Instance ID =

Filter

Launch Instance

Delete Instances

More Actions

Displaying 6 Items

<input type="checkbox"/>	Instance Name	Image Name	IP Address	Flavor	Key Pair	Status	Availability Zone	Task	Power State	Time since created	Actions
<input type="checkbox"/>	Cody-ubu18	CC-Ubuntu18.04	10.71.52.203	m1_small	vmec01	Active	nova	None	Running	1 day, 20 hours	Create Snapshot
<input type="checkbox"/>	Cody	CC-CentOS7	10.71.52.159 Floating IPs: 129.114.24.3	m1_medium	vmec01	Active	nova	None	Running	1 day, 22 hours	Associate Floating IP Attach Interface Detach Interface Edit Instance Attach Volume Detach Volume Update Metadata Edit Security Groups Edit Port Security Groups
<input type="checkbox"/>	train-code	CC-Ubuntu18.04	10.71.52.18 Floating IPs: 129.114.25.81	m1_medium	vmec01	Active	nova	None	Running	5 days, 20 hours	
sharednet1											

25.2. Work with KVM using the GUI

213

Edit Instance ✕

Information *Security Groups

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

Warning: If you change security groups here, the change will be applied to all interfaces of the instance. If you have multiple interfaces on this instance and apply different security groups per port, use "Edit Port Security Groups" action instead.

All Security Groups

Filter Q

vscode	+
newgroup	+
Allow ICMP	+
Allow SSH	+

Instance Security Groups

Filter Q

default	-
---------	----------------

CancelSave

25.2.7 Load Balancer as a Service

Available on **KVM@TACC** is the OpenStack Octavia Load Balancer as a Service (LBaaS). This service allows a single IP address to be used to distribute connections among a number of virtual machine instances. For the following description, it is assumed that there are already several virtual machines running an HTTP server on port 80, serving a page at the root path. To create a *Load Balancer*, click on *Project > Network > Load Balancers* in the navigation sidebar, then the *Create Load Balancer* button. This will open the *Create Load Balancer* dialog.

Create Load Balancer

Load Balancer Details *

Listener Details *

Pool Details *

Pool Members

Monitor Details *

Provide the details for the load balancer.

Name

IP address

Description

Flavor

Subnet *

Admin State Up

Yes

No

✕ Cancel

< Back

Next >

Create Load Balancer

Give your load balancer a name, and select the subnet that corresponds to the one used by the virtual machines. Click *Next*, or *Listener Details*.

The listener is the port that will accept incoming connections. Select the appropriate protocol for the service, in this case *HTTP*. If selecting *TCP* or *UDP* also provide the desired port. Click *Next* or *Pool Details*.

Choose the desired load balancing algorithm. This will determine the way in which the load balancer will select which VM receives incoming requests. Click *Next* or *Pool Members*.

Here you will select the virtual machines that will participate in the load balancing. Click the *Add* button next to the instances, after which their IP address and subnet will be added to the *Allocated Members* list at the top. You will need to provide the port number for the hosted service for each member. For our HTTP servers, it is port 80. This does not need to match the port of the load balancer's *listener*.

Once you've selected the pool members, click *Next* or *Monitor Details*. Here you will configure how the load balancer monitors the services on the virtual machines to ensure that they are ready to receive traffic. In our example, selecting *HTTP* adds configuration options for *HTTP Method*, *Expected Codes*, and *URL path*. Since the HTTP services on the VMs in the *pool members* are configured to serve a page on the root path, the default values will work. Click *Create Load Balancer*.

While the load balancer is being created, the dashboard will show a *Provisioning Status* of *Pending Create*. Once the process is complete, the status should be *Active*, and *Operating Status* should be *Online*. An *Operating Status* of *Offline* or *Error* indicates that the load balancer cannot satisfy the service check specified in *Monitor Details*. Ensure that the services are running on each VM, and that they return the expected status.

You can assign a Floating IP address to the load balancer by clicking on the down arrow button next to *Edit Load Balancer*, and selecting *Associate Floating IP*. This process is similar to associating a Floating IP to a virtual machine.

Create Load Balancer



Load Balancer Details *

Listener Details *

Pool Details *

Pool Members

Monitor Details *

Provide the details for the listener.

Name

Description

Protocol *

✓

HTTP

TCP

UDP

Port *

TCP Inspect Timeout

Member Connect Timeout

Member Data Timeout

Connection Limit *

Admin State Up

Yes

No

✕ Cancel

< Back

Next >

Create Load Balancer

Create Load Balancer



Load Balancer Details *

Listener Details *

Pool Details *

Pool Members

Monitor Details *

Provide the details for the pool.

Name

Description

Algorithm *

✓

LEAST_CONNECTIONS

ROUND_ROBIN

SOURCE_IP

Admin State Up

Yes

No

✕ Cancel

< Back

Next >

Create Load Balancer

Create Load Balancer

[Load Balancer Details *](#)[Listener Details *](#)[Pool Details *](#)[Pool Members](#)[Monitor Details *](#)

Add members to the load balancer pool.

▼ Allocated Members

IP Address	Subnet	Port	Weight
No members have been allocated			
			<button>Add external member</button>

▼ Available Instances

Q

http

Name ^

IP Address

http-1

10.71.52.144

Add

http-2

10.71.52.82

Add

http-3

10.71.52.245

Add

✕ Cancel< BackNext >Create Load Balancer

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	
> 10.71.52.144	Chameleon Ops-subnet	80	1	<button>Remove</button>
			<button>Add external member</button>	

▼ Available Instances

Q

http

Name ^

IP Address

http-1

10.71.52.144

Add

http-2

10.71.52.82

Add

http-3

10.71.52.245

Add

✕ Cancel< BackNext >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

Type *

HTTP

Max Retries Down *

3

Delay (sec) *

5

Max Retries *

3

Timeout (sec) *

5

HTTP Method

GET

Expected Codes

200

URL Path

/

Admin State Up

Yes

No

✕ Cancel

< Back

Next >

Create Load Balancer

Project / Network / Load Balancers

Load Balancers

Q Click here for filters or full text search.

✕

+ Create Load Balancer

Delete Load Balancers

Displaying 1 item

<input type="checkbox"/>	Name ^	IP Address	Operating Status	Provisioning Status	Admin State Up	
<input type="checkbox"/>	> lb-http	10.71.52.51	Offline	Pending Create	Yes	Edit Load Balancer

Project / Network / Load Balancers

Load Balancers

Q Click here for filters or full text search.

✕

+ Create Load Balancer

Delete Load Balancers

Displaying 1 item

<input type="checkbox"/>	Name ^	IP Address	Operating Status	Provisioning Status	Admin State Up	
<input type="checkbox"/>	> lb-http	10.71.52.51	Online	Active	Yes	Edit Load Balancer

218

Chapter 25. KVM

instance. Making changes to the various components of the load balancer by clicking on the blue-colored name of the load balancer in the list. From here, the *listeners*, *pools*, and *health monitors* can be updated, if needed.

To learn more about how to use the Octavia Load Balancer, refer to the [Basic Load Balancing Cookbook](#) on the official OpenStack documentation

25.3 Work with KVM using the CLI

For general information on CLI authentication and use, please see the [command-line-interface](#) section.

25.3.1 Uploading qcow2 images to raw format for better instance launch performance

KVM images are stored on our Ceph cluster, which is able to serve raw images much faster than qcow2 for instance launches. Openstack includes the experimental command Glance image-create-via-import, which allows uploading of images in various standard formats including qcow2 to then be automatically converted to raw in the backend.

In order to use this method, authenticate to KVM using the OpenStack RC script downloaded from the [KVM@TACC](#) site as described in *The OpenStack RC Script*.

Next, issue the following command:

```
glance image-create-via-import --container-format bare --disk-format qcow2 --
↪file </path/to/image> --name <image name>
```

Details and other options for this command are available via the Glance [image-create-via-import](#) documentation.

Attention: Glance image-create-via-import is currently unable to handle conversion of iso images to raw.

Alternatively, you may convert qcow2 images to raw format before upload. qemu-img is one tool that is able to this with the following command:

```
qemu-img convert -f qcow2 -O raw <original.qcow2> <converted.img>
```

Once converted, use glance to upload the image:

```
openstack image create --file </path/to/converted.img> --disk-format raw
↪<image-name>
```

Details and other options for this command are available within [Openstack](#) documentation.