
HNSciCloud Rhea Documentation

Release D-PIL-3.2 (2018-04-17)

HNSciCloud Consortium

May 07, 2018

Contents

1	Introduction	1
2	Getting Started	5
3	Researchers	13
4	Administrators	35
5	Data Coordinators	55
6	Support	73
7	Platform Services	75
8	Platform Demo	77
9	Platform Training	97
10	Consortium	119

1.1 HNSciCloud

Ten of Europe's leading public research organisations launched the [Helix Nebula Science Cloud \(HNSciCloud\) Pre-Commercial Procurement \(PCP\)](#) to establish a European hybrid cloud platform that will support the high-performance, data-intensive scientific use-cases of this "Buyers Group" and of the research sector at large.

This PCP calls for the design and implementation of hybrid innovative Infrastructure as a Service (IaaS) solutions for compute, storage, network connectivity, Federated Identity Management and Service Payment Models, to augment and to enhance the science community's existing systems.

The RHEA Group consortium's HNSciCloud design builds on a solid foundation of engineering expertise, existing open source software and commercial services:

- RHEA System Engineering & Cyber Security expertise
- SixSq's Nuvla, a SlipStream-based hybrid-cloud management service
- Cyfronet's OneData for Data Management
- Advania's IaaS Cloud Infrastructure (OpenStack)
- Exoscale IaaS Cloud Service (CloudStack)
- Open Telekom Cloud, T-Systems' IaaS Cloud Service (OpenStack)

The R&D innovations will be incorporated into these services as part of our commercial offerings, with minimum intrusion into the buyers' infrastructure, including:

- Peta-scale data management solution, portable to any cloud
- Flexible container management
- Single dashboard to better manage hybrid resources, including SLA compliance monitoring
- Unified accounting, integrating multi-cloud provider charges and billing
- Authentication from user to cloud providers

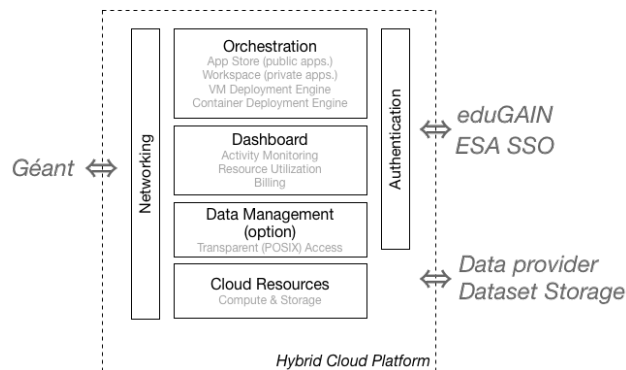
The Pilot (Phase 3) exercises the Consortium’s platform at a larger scale and allows for continued evolution of the platform to meet the needs of scientific computing.

1.2 Functional Overview

The hybrid cloud platform coming from the design phase provides cost-effective access to large-scale compute and storage resources from multiple providers. The solution brings together three commercial services, an authentication infrastructure that supports federated identities, and a data management infrastructure that can handle the large data sets of the Buyers Group. All the components are based on open source software released under the Apache 2.0 license, which allows liberal academic and commercial reuse.

Our solution includes the following main components:

- Authentication (KeyCloak): federates external identity providers, allowing users to use their “home” credentials to access the hybrid cloud platform
- Orchestration (Nuvla): allows users to manage the full lifecycle of cloud applications with a high degree of automation
- Data Management (Onedata + GlusterFS/S3): allows users to access and to manage large datasets hosted in hybrid cloud infrastructures and/or at a Buyers Group organization with minimum intrusion
- Networking (peering with GEANT): allows access to all the platform services from anywhere with enhanced access from sites connected to GEANT
- Cloud Resources (Advania, Exoscale, and OTC): IaaS and storage services accessible from the hybrid cloud platform
- Dashboard (Nuvla): provides an overview of Users’ current activities, resource utilisation, and costs.



The integration of these key components was demonstrated during the prototype phase. The pilot phase will concentrate on validating the platform at scale.

1.3 Actors

The primary users of the hybrid cloud platform will be researchers who want to analyze large datasets. However, there are many other actors involved to make the platform useful. To be as exact as possible when describing interactions with the platform, we have identified the full set of actors:

Researcher A person from a Buyers Group organization who analyzes scientific data by deploying instances of cloud applications (defined by Application Developers) for himself.

Application Operator A person from a Buyers Group organization who deploys and manages instances of cloud applications (defined by Application Developers) for others.

Data Service Operator A person from a Buyers Group organization or the Consortium who is responsible for deploying and maintaining the data services specific to an organization, project, or experiment.

Application Developer A person from a Buyers Group organization, Consortium or other organization who develops generalized software or services for use by others that use the platform's services, including data sets maintained by a Buyers Group organization. Defines (scalable) applications on the platform that can be deployed by a Researcher or Application Operator.

Data Coordinator A person from a Buyers Group organization who is responsible for managing the data (publishing, replicating, validating, archiving, etc.) for a specific organization, project, or experiment.

Account Coordinator A person from a Buyers Group organization who is responsible for managing the accounts (including credentials and quotas), monitoring resource utilization, and tracking costs.

Platform User or User A Researcher, Application Operator, Data Service Operator, Application Developer, Data Coordinator, Account Coordinator.

Broker Service Provider The organization that provides the cloud application management and brokering services for the platform, i.e. Nuvla.

Service Provider A “broker service provider” or “IaaS service provider”.

Consortium The organizations that together provide the hybrid cloud platform for HNSciCloud.

1.4 Scope and Coverage

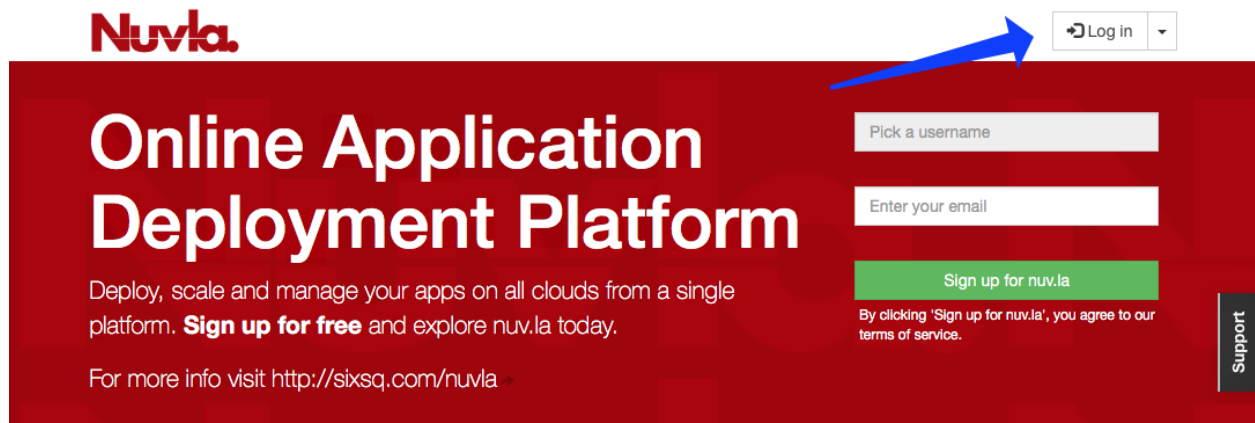
This documentation covers the essentials for learning about and getting started with the HNSciCloud hybrid cloud platform from the RHEA collaboration. It contains only information specific to the platform as a whole. Documentation for the individual services that comprise the platform are available elsewhere and may need to be consulted for anything other than simple use cases. Links to that documentation are provided in the *Platform Services* section.

In order to start using the system, you will need to setup and configure your accounts. This section describes how to do this and then how to ensure that everything is working correctly.

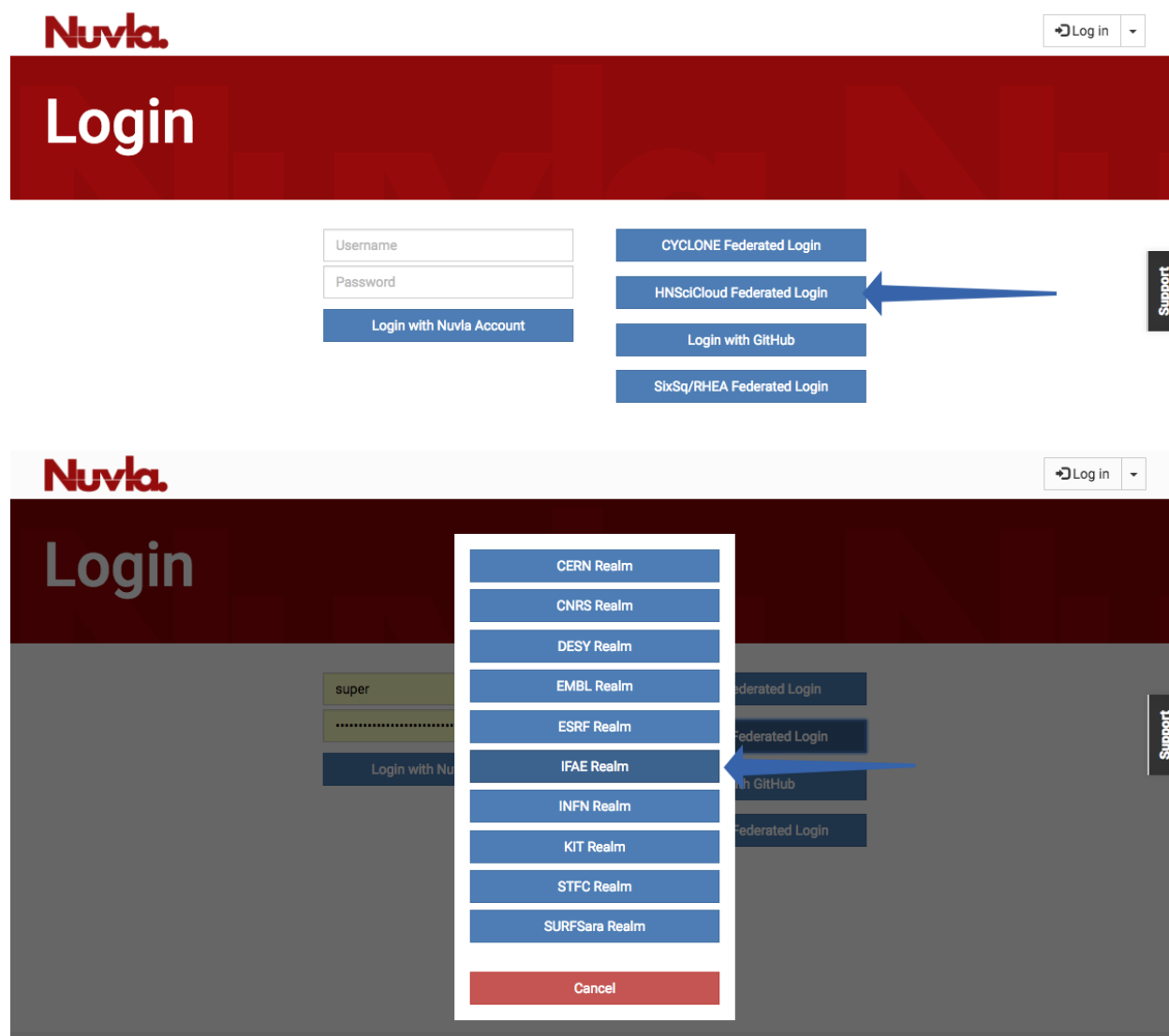
2.1 Account Activation

The main interface for managing cloud resources is [Nuvla](#). New users may create their accounts by simply signing into Nuvla with their institutional credentials (through the eduGAIN and Elixir AAI identity federations). The full procedure to activate an account in Nuvla using your institutional credentials is as follows:

1. Click on the login button which will then take you to a page to select your login method.

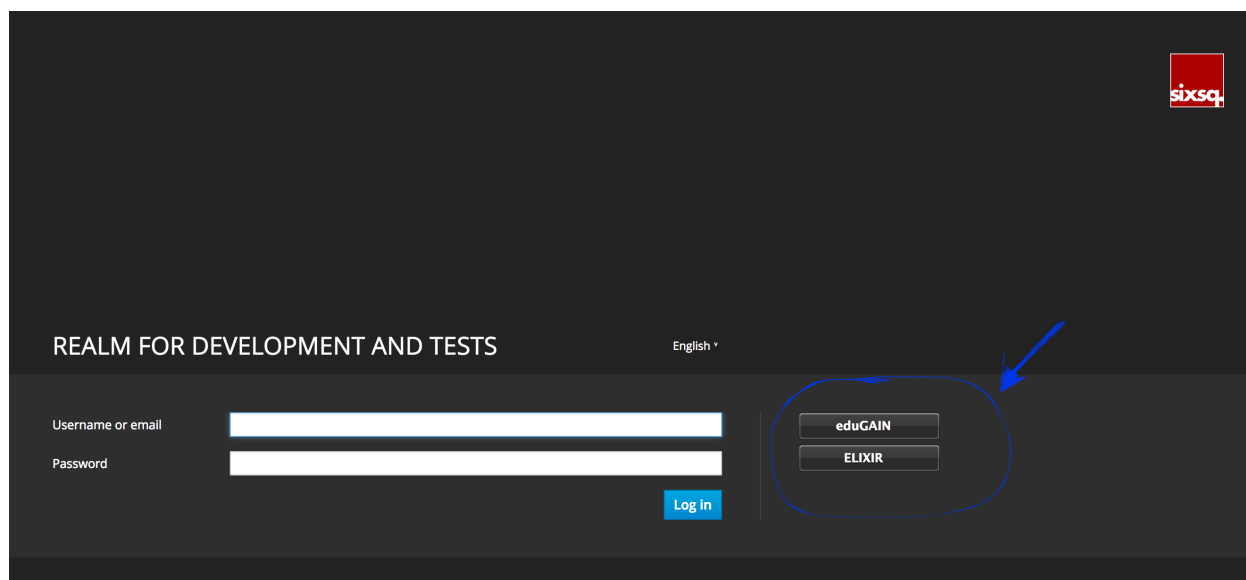


2. On this page, HNSciCloud users shall select “*HNSciCloud Federated Login*” and select their realm (or tenant), as shown in the figure below.



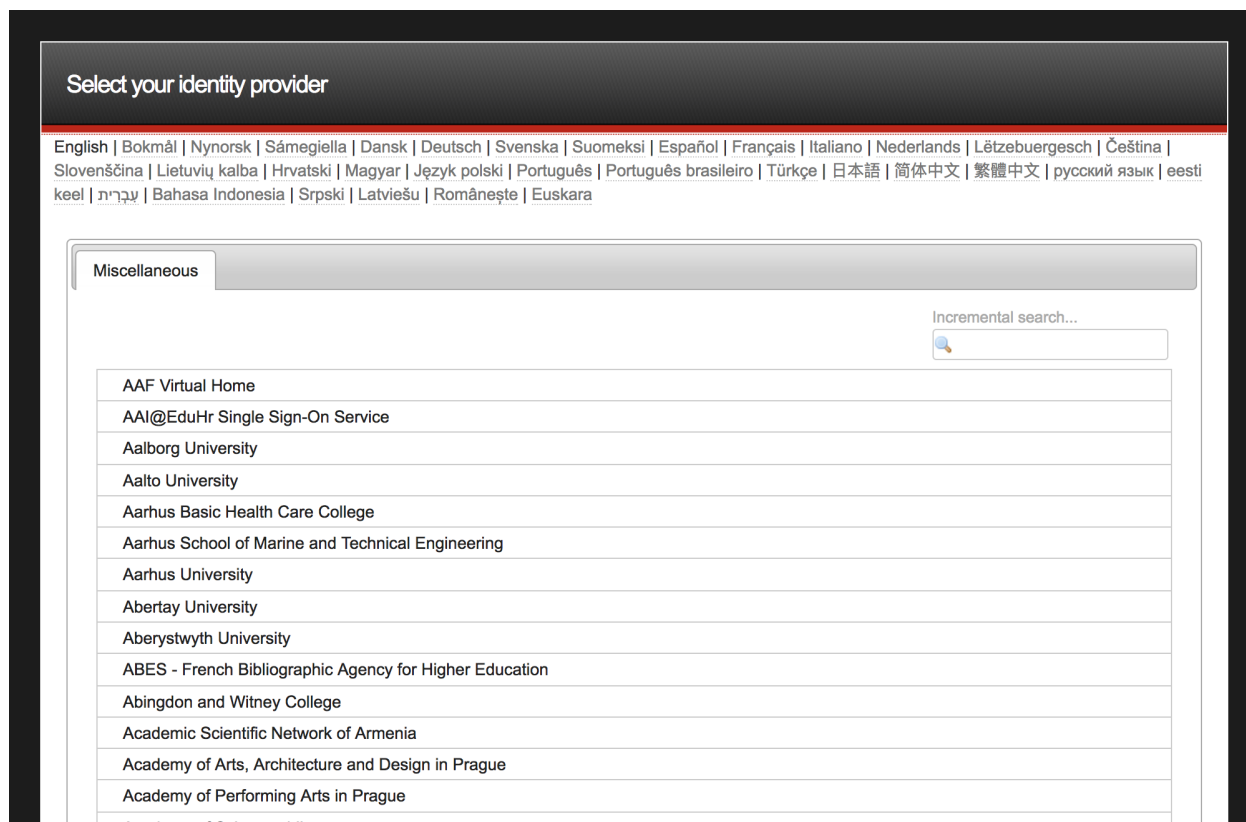
This will redirect users to their respective login realm in [SixSq's Federated Identity Portal](#). This portal is SixSq's authentication and authorization infrastructure (AAI) and it uses [Keycloak](#) and [simpleSAMLphp](#) underneath in order to make the authentication bridge between client applications (like Nuvla) and identity federations like eduGAIN and ELIXIR AAI (using SAML2.0).

3. Users shall then select which identity federation they want to use, either eduGAIN or ELIXIR.



- For both eduGAIN and ELIXIR, users will then be presented with a comprehensive list of identity providers and a search field.

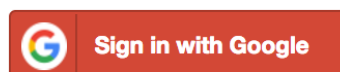
eduGAIN:



ELIXIR:



Log in using ...



or

your institutional account

Aalborg University
Aalto University
Aarhus Basic Health Care College
Aarhus School of Marine and Technical Engineering
Aarhus University
ACOnet staff
Alexander Fleming Biomedical Sciences Research Center
Alexander Stuart test IdP (vin)

Upon selection of the identity provider, users will be redirected to their institute's login page.

5. When successfully authenticating with the identity provider, the user will then be redirected back to Nuvla, which will show the active session parameters, including the user's full username and roles. [This page](#) can be viewed at anytime to show this information.

Nuvla

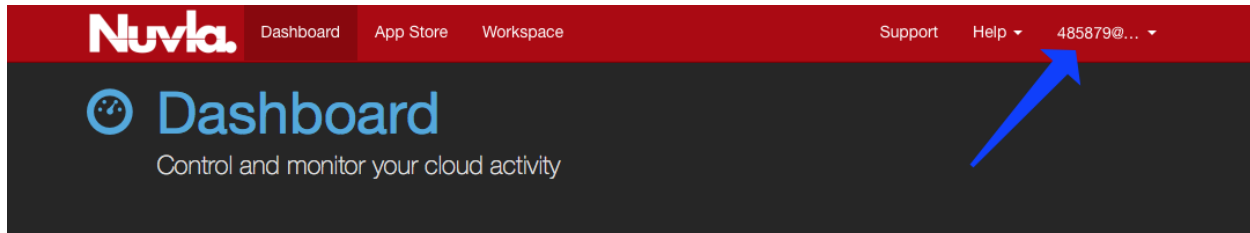
Log in

Login

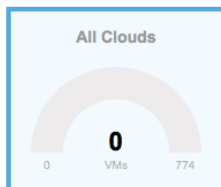
Redirecting...

clientIP	83.172.201.123
expiry	Fri, 21 Jul 2017 08:47:50 UTC
id	session/2d1fcca0-0449-4c7f-bacb-35e8765652fd
method	oidc
redirectURI	/authn/login
roles	session/2d1fcca0-0449-4c7f-bacb-35e8765652fd USER ANON SixSq:uma_authorization SixSq:offline_access
server	nuv.la
sessionTemplate	{href "session-template/sixsq"}
username	485879@vho-switchaai.chhttps://aai-logon.vho-switchaai.ch/idp/shibboleth!https://fed-id.nuv.la/samlbridge/module.php/saml/sp/metadata.php/sixsq-saml-bridge:u

6. The user will then be automatically redirect to the Nuvla dashboard.



Usage

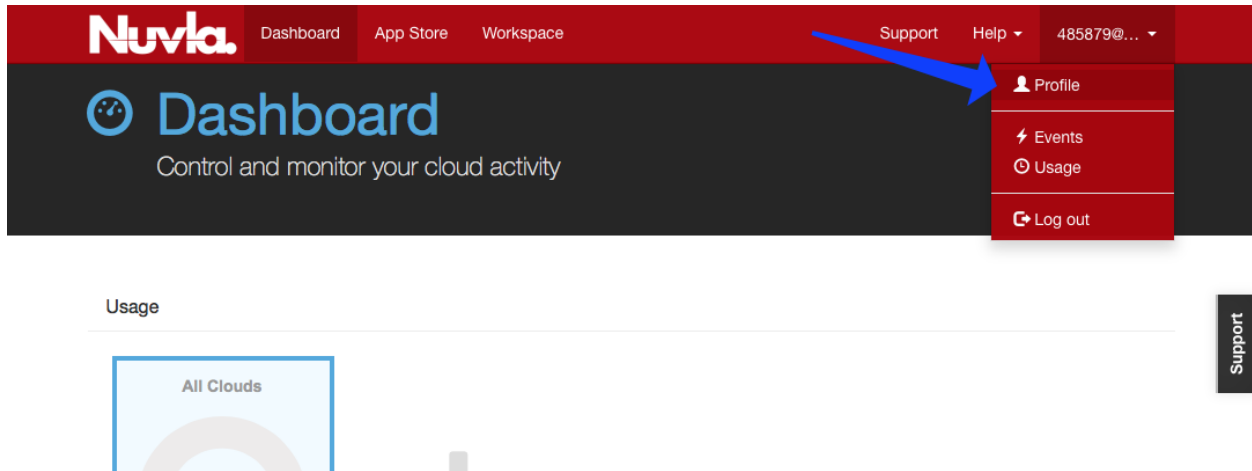


Support

7. If users already have an active session, then they'll be automatically redirected to Nuvla, without going to the identity provider's login page.
8. At this point, new users have been automatically registered in Nuvla and their accounts are now active.
9. Users that are an **ACCOUNT MANAGER** must send an email to support@sixsq.com asking *admin* rights to the tenant, which shall be granted by SixSq, in SixSq's [Federated Identity Portal](#), where the account managers can then manage users, groups and roles (as described in [here](#)).
10. All **OTHER USERS** must contact the account manager for the realm so that the manager can assign roles to them or add them to a group.

2.2 Account Configuration

To use Nuvla to provision the data management services or cloud applications on the IaaS cloud infrastructures, you must configure your Nuvla account. To access your user profile, click on “Profile” link under your username.



To update your user profile, click on the “Edit...” on the right side below the page header.

2.2.1 Remote Machine Access

To allow you have remote access to the (Linux) virtual machines that you deploy, you should provide a public SSH key. Once this key has been added to your profile, Nuvla will automatically configure all deployed virtual machines with this key, giving you ‘root’ access to your deployed machines. The instructions for creating an SSH key pair and configuring your profile can be found in the [Remote Machine Access](#) section of the SlipStream documentation. This documentation also describes the installation of a “Remote Desktop Connection” client for accessing Windows machines.

2.2.2 Cloud Credentials

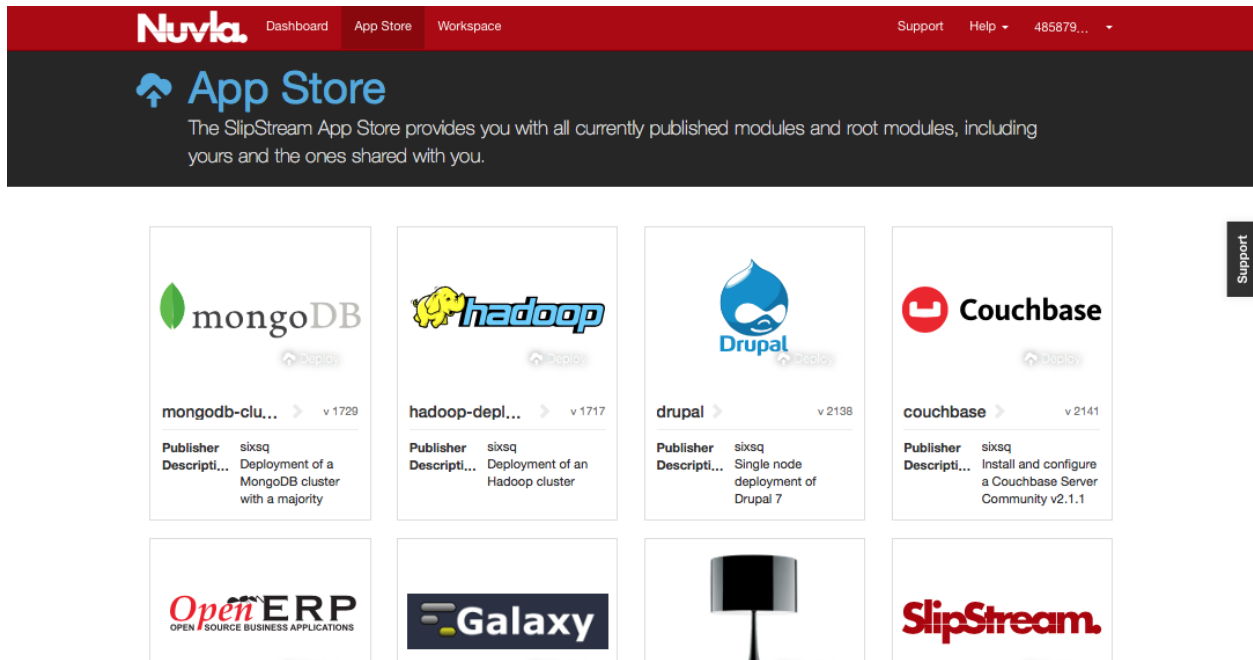
In order to be granted access to the OTC, Exoscale and Advania cloud credentials, technical users must contact their account managers, asking for a specific user role (**can_deploy**) to be given to them, as described in [Cloud Provider Configuration](#).

2.3 Components and Applications

The first place to look for interesting components (single virtual machine services) and applications (multiple machine services) is the [Nuvla App Store](#).

Within the [Nuvla Workspace](#), there are other applications of interest:

- `examples/images`: Minimal distributions of common operating systems. Usually used as the basis for other components.
- `apps`: Curated list of applications that can be used as examples for your own applications.
- **HNSciCloud**: [This](#) workspace contains several prearranged components and applications to facilitate the testing and evaluation process, including for example:



- HNSciCloud/Benchmarking: Both generic and HNSciCloud-specific benchmarks for evaluating the system. Relevant for Test Cases 2.2, 5.1 and 11.4.3.
- HNSciCloud/Images: A subset of examples/images, containing only the HNSciCloud specific operating systems.
- HNSciCloud/VMProvisioningPersonalisation: An App for testing the provisioning and contextualization of a VM, according to Test Case 2.5.
- HNSciCloud/S3EndpointTest-Exoscale_OTC: An App for testing S3 in both Exoscale and OTC, according to Test Case 2.3.
- HNSciCloud/HDF5_IO: An App for verifying HDF5 compliance with the VMs' local storage in the cloud, according to Test Case 4.1.

Other application definitions will appear over time. If you have specific needs, contact SixSq support to request new ones.

This section contains howtos for common tasks that will be carried out by researchers.

3.1 IMPORTANT - Scaling Guidelines

In the past, some scalability issues have been observed when provisioning big deployment through Nuvla.

While this issue is being worked on, technical users are kindly asked to respect the following deployment guidelines in Nuvla.

3.1.1 Deploying Scalable Applications

When deploying a scalable application, users should **not scale up or down by more than 100 VMs at the time**. Also, users should **keep the total number of VMs per application deployment under 300**.

3.1.2 Deploying Single Components

For single component deployment, **each user is able, for now, to deploy up to 500 instances**.

3.2 Start a Linux VM

The main interface for managing virtual machine deployments is [Nuvla](#). The procedure to deploy standard Linux-based VMs is straightforward:

- Navigate to the component/image you want to deploy,
- Click on the “deploy” button,
- Choose the cloud/offer to use in the deployment dialog, and
- Provision the image by clicking the “deploy” button.

You will then be taken to the dashboard where you can follow the progress of your deployment. Detailed documentation can be found on the SlipStream documentation website, specifically the [deployment documentation](#).

Once the deployment reaches the “ready” state, you can log into your virtual machine via SSH. Be sure that you have configured your Nuvla account with your public SSH key.

The common “native” images that are supported across clouds can be found in the `examples/images` project within the Workspace. In general, Ubuntu 14/16, Debian 7/8, and CentOS 7 are well supported across clouds.

Other Linux distributions may be supported as well. Either you can ask SixSq (through support) to extend the list of available images or create your own [native image](#) component.

3.3 Start a Windows VM

The main interface for managing deployments is [Nuvla](#). Deploying Windows VMs is straightforward and the same as for Linux VMs:

- Navigate to the component/image you want to deploy,
- Click on the “deploy” button,
- Choose the cloud/offer to use in the deployment dialog, and
- Provision the image by clicking the “deploy” button.

You will then be taken to the dashboard where you can follow the progress of your deployment. Detailed documentation can be found on the SlipStream documentation website, specifically the [deployment documentation](#).

Once the deployment reaches the “ready” state, you can log into your virtual machine via Microsoft Remote Desktop.

The supported Windows images can be found in the `examples/images` project within the Workspace.

3.4 Deploy Docker Containers

[Docker](#) is a convenient system for defining, packaging, and deploying applications. It uses container technologies to allow portability between operating systems and to achieve fast start-up times. By default it will use images from the [Docker Hub](#), an open registry of containers.

On typical IaaS cloud infrastructures, you must first deploy a virtual machine, install Docker ([Docker Compose](#)), and then start your container. SixSq provides a Docker Compose recipe on [Nuvla](#) to make the installation of software and the deployment of containers easy.

To launch the Docker Compose virtual machine, find the [Docker Compose Component](#) either in the App Store or the Workspace (or by clicking the link!).

- Click on “Deploy”
- Choose the cloud you want to use, and
- Wait for the virtual machine to start.

Once the machine is ready, you can log into the machine via SSH using the Service URL link or manually. Once you are on the machine, you can then use Docker and Docker Compose as you normally would.

For example, a simple “Hello World” example:

```
$ docker run hello-world
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
```



```
b04784fba78d: Pull complete
Digest: sha256:f3b3b28a45160805bb16542c9531888519430e9e6d6fffc09d72261b0d26ff74f
Status: Downloaded newer image for hello-world:latest

Hello from Docker!
This message shows that your installation appears to be working correctly.

...
```

This will show you a message and indicate that the installation is working correctly. It will also provide some pointers for doing more complicated (and useful) tasks.

For example, you can try something similar from the [Docker Compose Getting Started](#) page. Following the instructions there, you can deploy a simple web application that provides a welcome message with a counter of how many times it has been loaded.

```
~/composetest$ docker-compose up
Creating network "composetest_default" with the default driver
Building web
Step 1/5 : FROM python:3.4-alpine
3.4-alpine: Pulling from library/python
acb474fa8956: Pull complete

...

redis_1 | 1:M 26 Jun 07:32:59.588 * The server is now ready to accept connections on
↪port 6379
web_1   | * Running on http://0.0.0.0:5000/ (Press CTRL+C to quit)
web_1   | * Restarting with stat
web_1   | * Debugger is active!
web_1   | * Debugger PIN: 241-972-540
```

Note: Instead of using `localhost` or `0.0.0.0`, you will need to use the **IP address of the virtual machine** (with the 5000 port!) from your remote browser. If everything worked correctly, you should see a message like “Hello World! I have been seen 2 times.”.

From here you might want to look at the entries [Deploy Docker Swarm](#) or [Deploy a Kubernetes Cluster](#).

3.5 Deploy Docker Swarm

Running Docker in “**Swarm**” mode allows you to deploy and control a cluster of machines running the Docker Engine. The Docker Swarm component in [Nuvla](#) automates the installation, configuration, and deployment of a swarm.

The [Docker Swarm component](#) can be found in the Workspace or App Store on Nuvla (or by clicking the link!). After finding the component, you can deploy a swarm by:

- Clicking on “deploy”,
- Choosing whether the deployment is [scalable](#),
- Choosing the number of masters and workers, and
- Launching the swarm by clicking on “Deploy” in the deployment dialog.

Once the deployment is ready, you can check the status of the swarm with:

```
$ docker info

...

Swarm: active
NodeID: ra0mpoiy6pnv7j6xnlbav3e12
Is Manager: true
ClusterID: 8gux86j2845pzs9uuwftgqnfu
Managers: 1
Nodes: 4

...
```

If you are logged into a manager, you should see that the “Is Manager” flag is true. Here a 5 node deployment was started with 1 manager and 4 workers.

You can get more details about the swarm by looking at the node status from a manager machine:

```
$ docker node ls
```

ID	AVAILABILITY	MANAGER STATUS	HOSTNAME	STATUS
o476smd7wm3s97rin0u6nn84q	↪Active		worker2310ef996-af16-4815-8ba7-c88d630d95f4	Ready
ra0mpoiy6pnv7j6xnlbav3e12	↪Active	Leader	master1310ef996-af16-4815-8ba7-c88d630d95f4	Ready
vl3c4ypaguwglypr5qko6zkgu	↪Active		worker3310ef996-af16-4815-8ba7-c88d630d95f4	Ready
vvz3beena6t4200bhag331d5b	↪Active		worker1310ef996-af16-4815-8ba7-c88d630d95f4	Ready

You should see a listing like the one above, if everything has worked correctly. The cluster is ready to be used.

To deploy a service to the swarm, you can follow the [docker swarm service](#) tutorial. From the manager node, you can deploy, list, inspect and remove the services as follows:

```
$ docker service create --replicas 1 --name helloworld alpine ping docker.com

gzhyxwm0jp4ddesv56g9gcv7

$ docker service inspect --pretty gzhyxwm0jp4d
```

```
ID:          gzhyxwm0jp4ddesv56g9gcv7
Name:        helloworld
Service Mode: Replicated
  Replicas:  1
Placement:
UpdateConfig:
  Parallelism: 1
  On failure:  pause
  Max failure ratio: 0
ContainerSpec:
  Image:       ↪
  ↪alpine:latest@sha256:b09306f2dfa3c9b626006b2f1ceeeaa6fcbfac6037d18e9d0f1d407260cb0880
  Args:        ping docker.com
Resources:
Endpoint Mode: vip

$ docker service ps gzhyxwm0jp4d
```

```

ID                NAME                IMAGE                NODE
↪ DESIRED STATE  CURRENT STATE      ERROR    PORTS
x9twksry8knc    helloworld.1    alpine:latest    master1310ef996-af16-4815-8ba7-
↪ c88d630d95f4    Running          Running about a minute ago

$ docker service rm gzhwxm0jp4d
gzhwxm0jp4d

```

See the Docker [Swarm](#) documentation for scaling and other management actions for your Docker applications.

3.6 Deploy a Kubernetes Cluster

[Kubernetes](#) is an open-source system for automating deployment, scaling, and management of containerized applications. With [Nuvla](#), you can deploy your own Kubernetes cluster automatically.

The [Kubernetes component](#) can be found in the Workspace or App Store on Nuvla (or by clicking the link!). After finding the component, you can deploy a cluster by:

- Clicking on “deploy”,
- Choosing whether the deployment is [scalable](#),
- Choosing the number of workers, and
- Launching the cluster by clicking on “Deploy” in the deployment dialog.

Once the cluster is ready, you can SSH into the master node. From here, list all the nodes and check their current status with:

```

$ kubectl get nodes
NAME                STATUS    AGE
159.100.240.160    Ready    2m
159.100.240.193    Ready    2m
159.100.240.64     Ready    2m

```

You should see output similar to the above if all the nodes have been correctly deployed and configured.

After checking the status of the cluster, you can then use the `kubectl` command to deploy and control services on the Kubernetes cluster.

If you are unfamiliar with Kubernetes, you can follow the tutorials and look at the reference documentation on the [Kubernetes](#) website.

For a simple test deployment of [nginx](#) (a web server), create a file `nginx.json` with the following contents:

```

{
  "kind": "Pod",
  "apiVersion": "v1",
  "metadata": {
    "name": "nginx",
    "namespace": "default",
    "labels": {
      "name": "nginx"
    }
  },
  "spec": {
    "containers": [{
      "name": "nginx",

```

```
    "image": "nginx",
    "ports": [{"hostPort": 80, "containerPort": 80}]
  }
}
```

You can then deploy this web server with the command:

```
$ kubectl create -f nginx.json
pod "nginx" created
```

You can then discover what node is running the webserver with:

```
$ kubectl describe pod nginx

$ kubectl describe pod nginx | grep Node:
Node:                               159.100.240.64/159.100.240.64
```

In this case the server is running on the node 159.100.240.64. You can point a browser to this node and then you should see the nginx welcome page.

Afterwards, you can run the command:

```
$ kubectl delete pod nginx
pod "nginx" deleted
```

to remove the nginx deployment from the cluster.

3.7 Monitoring

Nuvla provides a dedicated API resource for [monitoring cloud activities](#).

Refreshed continuously (~every minute), data gets collected from all the configured clouds. The monitored data is then assembled to include for each virtual machines, the following:

- vCPU
- RAM
- Disk
- Service offer

This information is collected for both cloud resources provisioned via Nuvla and provisioned directly to the different clouds (e.g. API, portal, CLI), by-passing Nuvla. This means that Nuvla provides a unified global view over all compute resources deployed, independently of it being used as the deployment engine or not.

Further, the monitoring resource inherits the basic CIMI functionality, which means it contains specific and precise ACLs, such that only specific users, groups and organisation members with the appropriate rights have access to this information.

This resource is key for other new features, such as the [Accounting](#) feature, as well as the extended [Quota](#) functionality.

The REST resource providing this functionality is called *virtual-machine* and can be used, for example, to query the following information:

- Count all virtual machines
- Count all virtual machines belonging to a give user (requires privileged rights)

- Count all virtual machines launched directly with the cloud, by-passing Nuvla
- Simulate what virtual machines a given user sees (requires privileged rights)
- Group currently running virtual machines, by billing period (e.g. billed by minute vs hour)
- List all virtual machines belonging to a given deployment
- Count of running virtual machines, grouped by cloud

From the list above, *privileged rights* include administration rights (super user), as well as group and organisation owners.

3.8 Accounting

Nuvla provides a new [dedicated API](#) resource for accounting.

The new feature regularly snapshots the global state of deployed resources provided by the [Monitoring](#) resource, to build a historical view of usage. In the process, it also pulls other valuable information such as pricing from the corresponding service offers.

This functionality, coupled with the extensive CIMI query and filtering capabilities, provides together a powerful, yet simple, way to extract accounting information from Nuvla.

The [dedicated API](#) documentation provides several examples of queries in order to extract the following type of queries, such as:

- Pricing for any from/to time interval (full days)
- Extract daily, weekly and monthly consumption aggregation (e.g. CPU, RAM, Disk)
- Extract peak usage over a given time interval, grouped by cloud, for a user, group and/or organisation
- Sum CPU, RAM and/or disk usage, for a given time interval and for a cloud, a user, group and/or organisation

Again, these are only example of possible queries. This feature can also be used to plot trends, trigger alerts and much more.

3.9 Quota

Nuvla protects users from overuse, by applying a quota mechanism. Users can query their quota and request increases to their group or organisation managers.

When deploying new resources, Nuvla checks if the requested deployment will see the user exceeding the quota. Only if this is not the case, will the deployment be authorised.

The quota now includes the following artefacts:

- CPU
- RAM
- Disk

For more details on how to use this feature can be found in the [monitoring cloud activities](#).

Note: This feature is very flexible and will be upgraded to include other monitored artefacts such as storage (e.g. object store).

3.10 Data Access

To facilitate a seamless access to the Buyers' distributed data sets, the project uses [Onedata](#) global data management system.

Onedata infrastructure, spanning multiple clouds, consists of two components - Onezone and Oneprovider, and user level Oneclient tool for the data access. Each Buyer Group has a single Onezone instance and per cloud instance of Oneprovider. Oneproviders attach to the Cloud's data stores and connect to Onezone, with the latter being the main entry point and enabling the global authentication and metadata management.

At this time, it is assumed that the distributed Onedata infrastructure is already deployed by the Buyer's Group [Data Coordinators](#) and the endpoints of Onezone and Oneproviders are available.

3.10.1 Using Nuvla to provision data access client

Users access their data on VMs using POSIX. The data becomes available on the VMs via mounting the required data sources with the help of [Oneclient tool](#). When deploying VMs via [Nuvla](#) service, users should use or, when building their own components, inherit from *oneclient-<OS>* components, which are available at <https://nuv.la/module/HNSciCloud/onedata>.

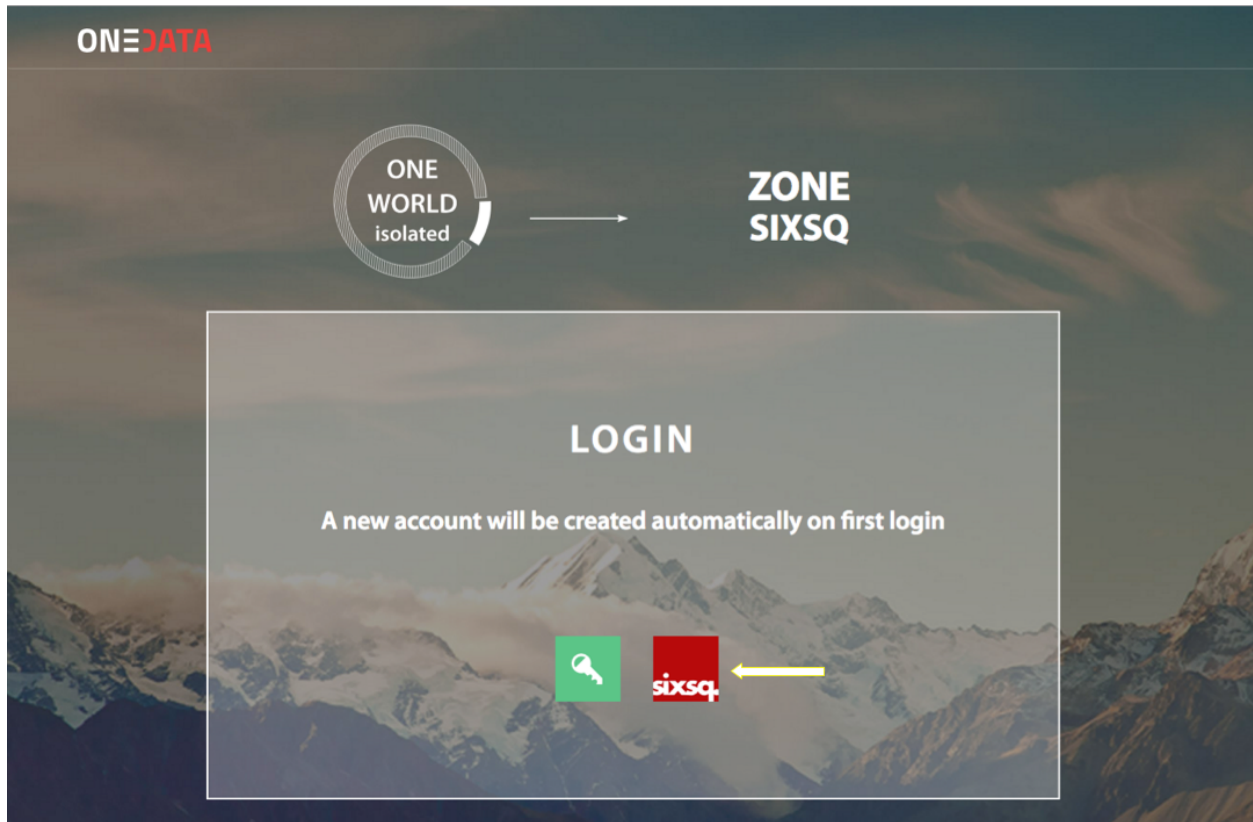
At the moment, for *oneclinet* to mount the data volumes on VMs and enable POSIX access to the data, users need to provide it with a data access token and the Cloud local Oneprovider endpoint.

Next versions of the platform will be extended to contain the auto-discovery of the Cloud local Oneprovider as well as auto-generation of the data access token.

Obtaining Onedata data Access Token

The following steps are required to get the data access token

- obtain the Buyer's Group Onezone endpoint
- in Onezone, authenticate via FedIdP



- when logged in Onezone web user interface, press *Access Tokens* in the top menu and press *Create new access token* button, then click on the *copy* pictogram

ONEDATA

GO TO YOUR FILES

DATA SPACE MANAGEMENT

+

Create new space

JOIN

Join a space

You can also become a provider yourself and support your own space.

GROUP MANAGEMENT

JOIN

Join a group

ACCESS TOKENS

Generate new token to get access to Onedata using command line client or REST API

MDAxNWxvY2F0aW9uIG9uZXpvcn

+

Create new access token

AUTHENTICATION SETTINGS

USER ALIAS

MANAGE ACCO

First

1

2

3

Get

To stor
least o

>

22

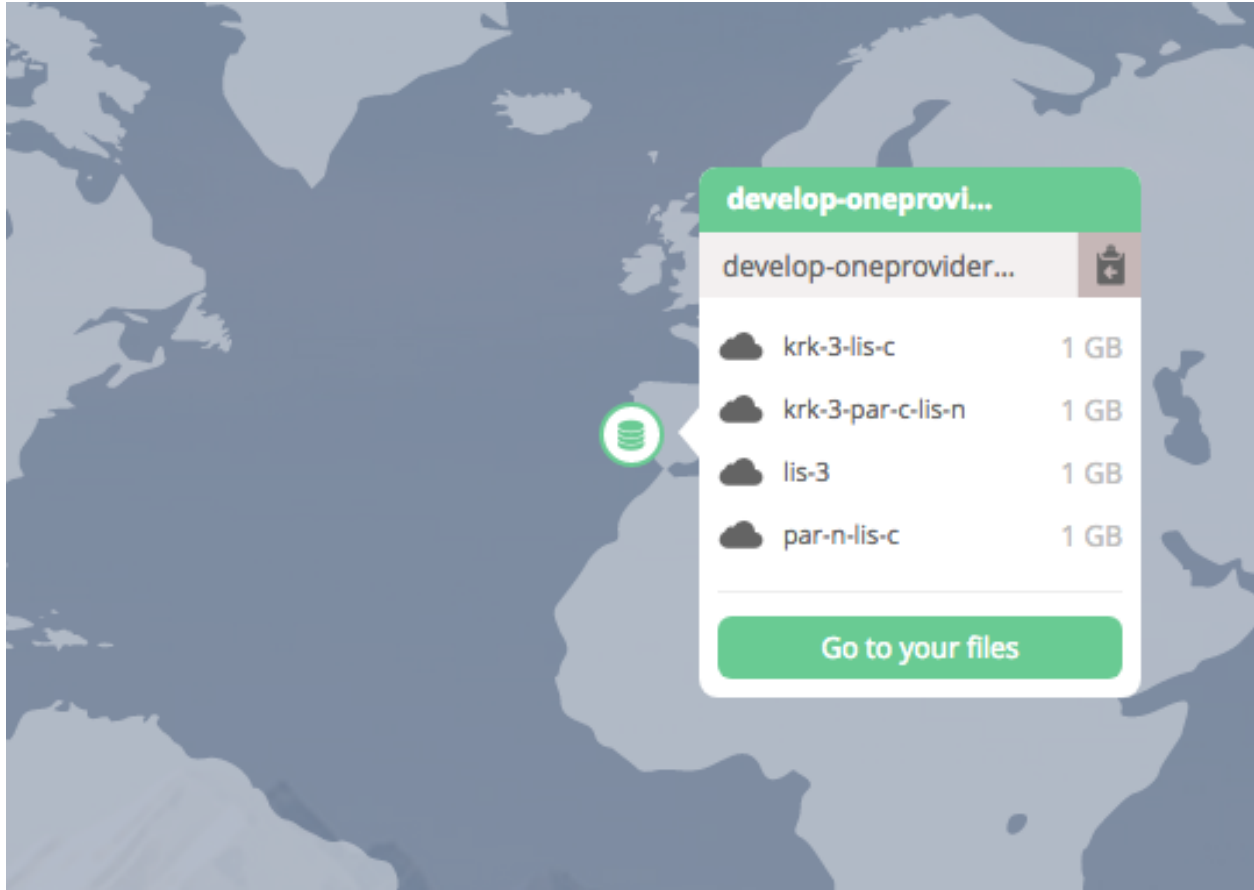
Chapter 3. Researchers

The copied data access token should be used with the **access-token** parameter in the *oneclient* component deployment.

Obtaining Oneprovider endpoint

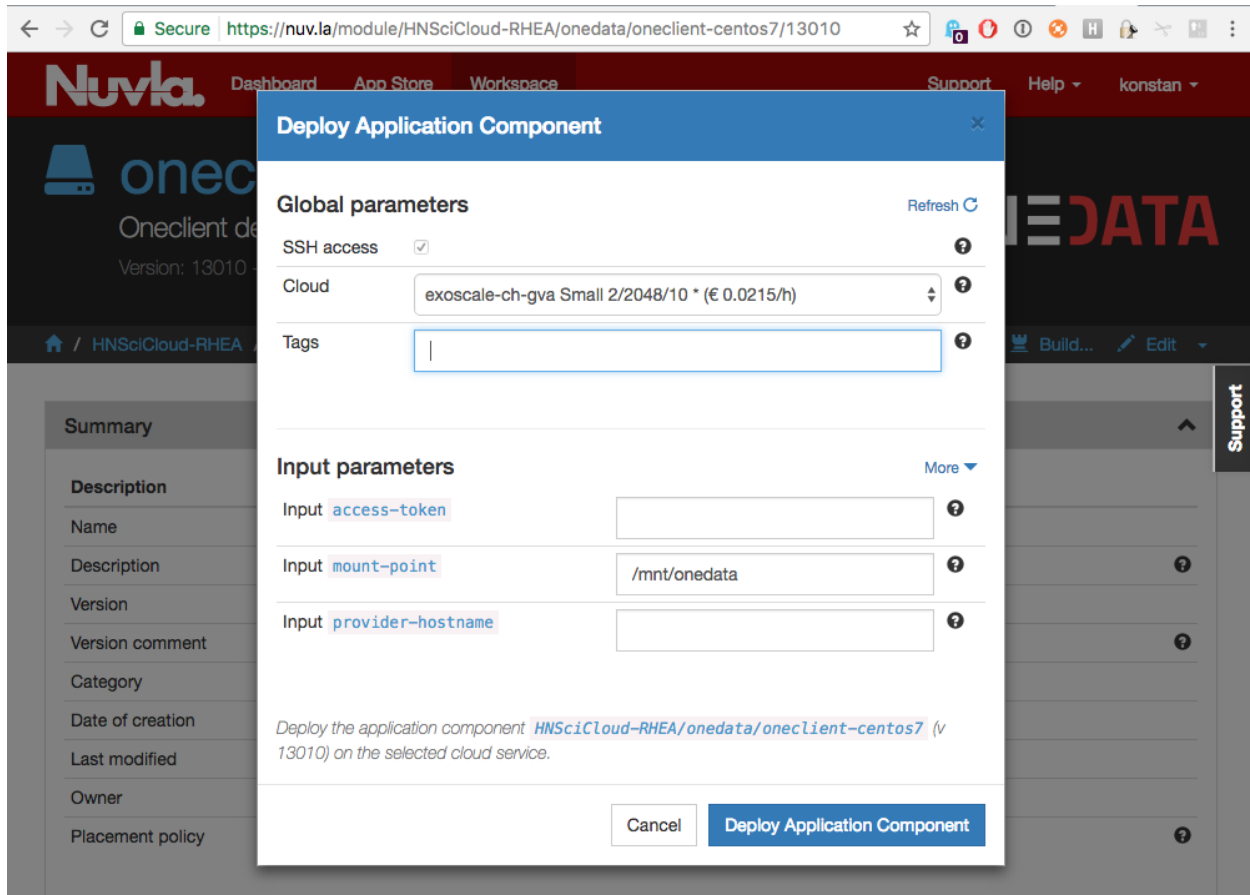
Check with your Data Manger for the IP/DNS name of the Oneprovider endpoint on the Cloud of your choice.

Oneprovider IP can also be easily copied from the Onezone ‘world’ view by clicking on the icon of selected One-provider instance and pressing *copy* button in the top right corner of the popup:



Provision VM for data access

Here it is explained on the example of CentOS 7 image. Go to <https://nuv.la/module/HNSciCloud/onedata/oneclient-centos7> and click on *Deploy*.



Select the Cloud on which you want to deploy the client via *Cloud* drop-down. Provide **access-token** and **provider-hostname** parameters. Optionally change the default mount point of the spaces in **mount-point** parameter. Then, click *Deploy Application Component* button.

3.10.2 Accessing data on VM

After the successful deployment, user should expect the spaces (supported by the selected provider) with the files available for POSIX access under **mount-point**. Example

```
# ls -l /mnt/onedata/
space-s3
space-gluster
# ls -l /mnt/onedata/space-s3
test.txt
#
```

3.10.3 External documentation

More information on how to

- access, manage and share your data;
- create groups of users with fine grained access rights;
- share and collaborate on your data

can be found in Onedata [User quickstart](#) and the [User guide](#).

3.11 Create Components and Applications

SlipStream (and Nuvla) allow you to define portable cloud components and applications. For SlipStream “components” are single virtual machine services and “applications” are multiple virtual machine services make up of one or more components.

To ensure that the components and applications are portable, SlipStream uses an application model that described customized images as a set of deltas (recipes) that describe the changes from “native images”.

In other words, components define a template, including resources characteristics such as default CPU, RAM and disk. They also define recipes (aka scripts) that are executed as part of the deployment, to transform the image into a fully configured VM. The native images are normally minimal operating system images that are have consistent behavior across clouds.

The SlipStream tutorial contains an [extensive section](#) on how to create your own components and applications. Please refer to that tutorial. For cases where quick start up times are important, you can also [build images](#) on clouds that support it.

3.12 Binary VM Images

Uploading of binary VM images is strongly discouraged. Instead, use of the Nuvla features for creating [portable recipes](#) (optionally with the [build feature](#)) or the use of containers are preferred solutions.

Note: From experience, **native base images provided by cloud providers are optimised for their cloud**. They tend to boot faster, are smaller and ultimately are more performant. Custom binary images tend to under perform compared to native images, and generally require adjustments, which can be time consuming.

3.12.1 Open Telkom Cloud (OTC)

Nonetheless, if you must use a binary VM image, you can upload the image to OTC. The documentation for making private images available on OTC via the GUI is available from the OTC website:

- [Uploading an External Image File](#)
- [Registering the Image File as a Private Image](#)
- [Creating an ECS Using an Image](#)

The API can also be used. The documentation for this is:

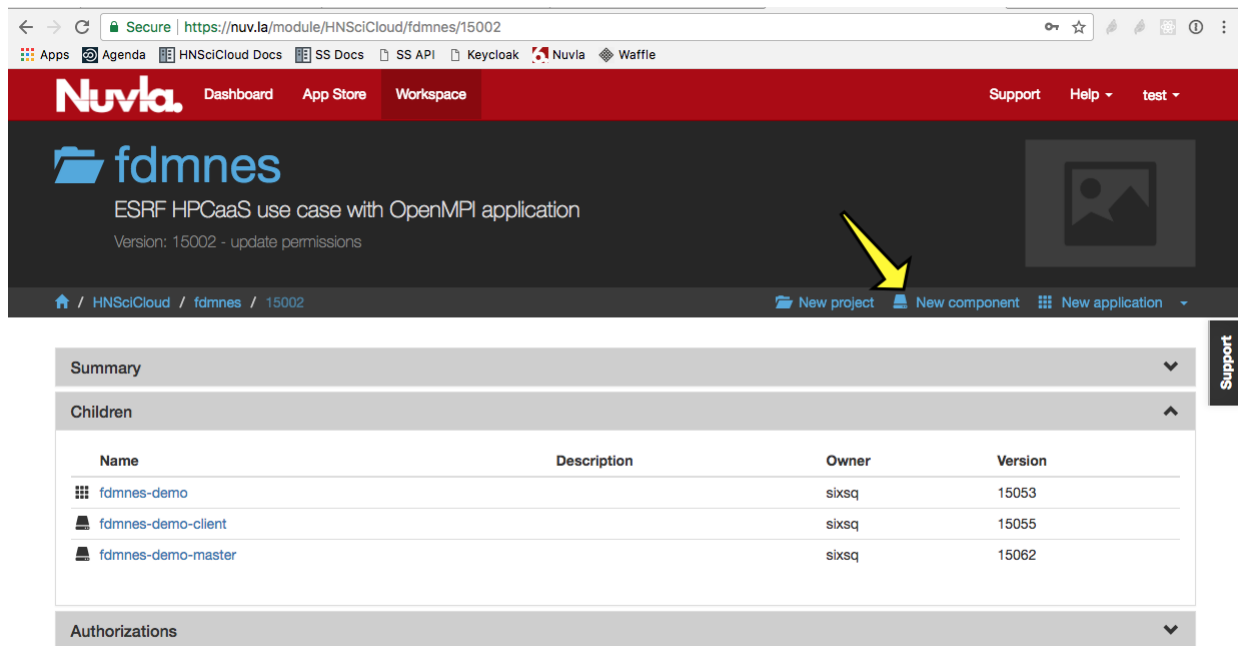
- [Uploading an Image via API](#)
- [Registering an Image File as a Private Image via API](#)

You can then use this image directly from the OTC web browser interface and the API.

3.12.2 Integration with Nuvla

If you want to use this image from Nuvla, you must define it as a [Native Image](#). This requires that [cloud-init](#) be installed in the image (along with its Python dependency). Once defined as a native image, it can be used like any other SlipStream component.

The detailed process is described below. First, navigate to the project (subdirectory) where you want to create your native component (image) and click on the “New component” link.

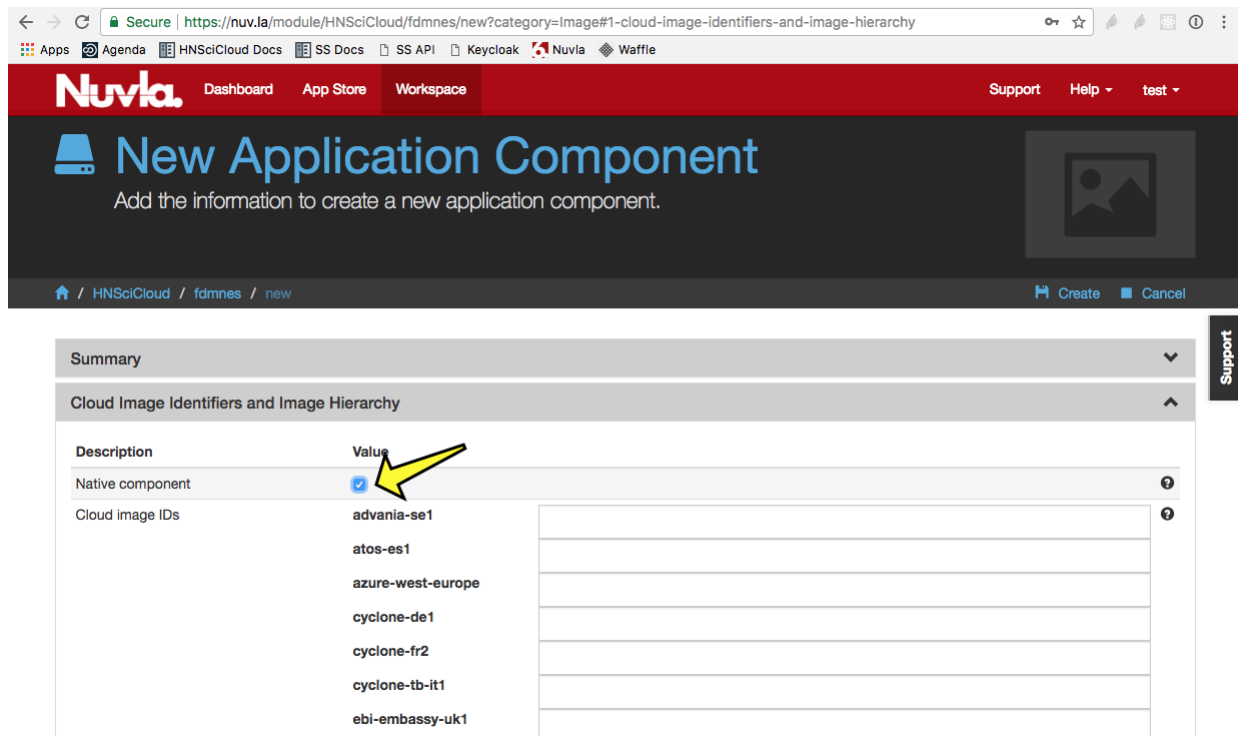


The screenshot shows the Nuvla dashboard for the 'fdmnes' project. The breadcrumb trail is: Home / HNSciCloud / fdmnes / 15002. In the bottom right of the dashboard, there are three links: 'New project', 'New component' (highlighted with a yellow arrow), and 'New application'. Below the dashboard, there is a 'Summary' section and a 'Children' table.

Name	Description	Owner	Version
fdmnes-demo		sixsq	15053
fdmnes-demo-client		sixsq	15055
fdmnes-demo-master		sixsq	15062

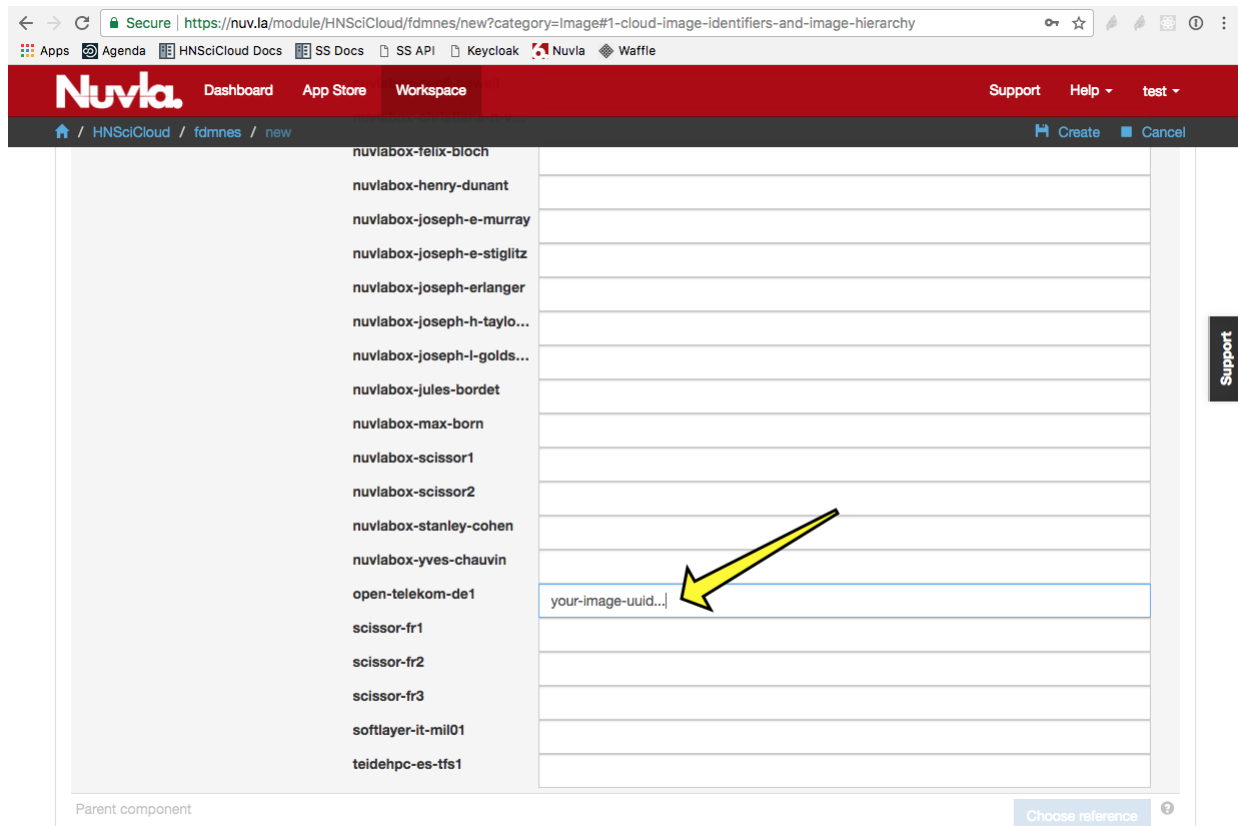
Below the table is an 'Authorizations' section.

To make this a native component (i.e. one which references a binary image in the targeted cloud), check the “Native component” checkbox. Provide the image identifier(s) for the cloud(s). The image shows an example for OTC.

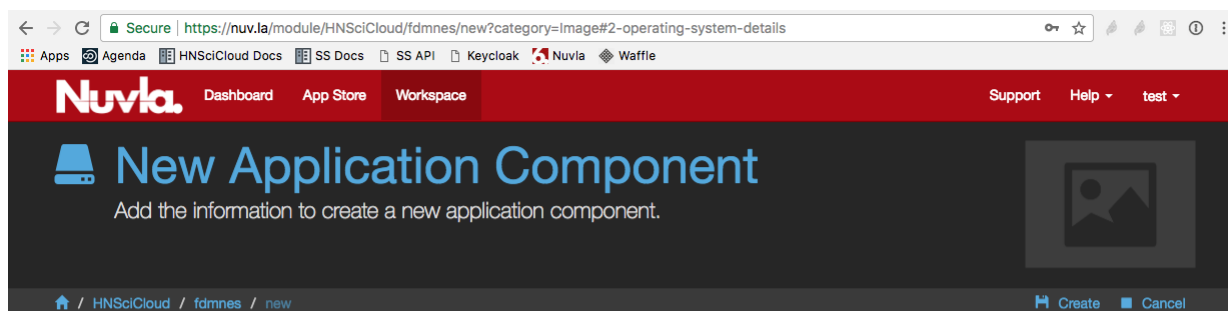


The screenshot shows the 'New Application Component' form. The breadcrumb trail is: Home / HNSciCloud / fdmnes / new. The form has a 'Summary' section and a 'Cloud Image Identifiers and Image Hierarchy' section. In the 'Cloud Image Identifiers and Image Hierarchy' section, there is a table with 'Description' and 'Value' columns. The 'Native component' checkbox is checked, and a yellow arrow points to it.

Description	Value
Native component	<input checked="" type="checkbox"/>
Cloud image IDs	<div> <div>advania-se1</div> <div>atos-es1</div> <div>azure-west-europe</div> <div>cyclone-de1</div> <div>cyclone-fr2</div> <div>cyclone-tb-it1</div> <div>ebi-embassy-uk1</div> </div>



You will also want to configure the image's operating system details and the minimum resources required for the image.



Description	Value
Platform	CentOS
Login username	

Summary

Cloud Image Identifiers and Image Hierarchy

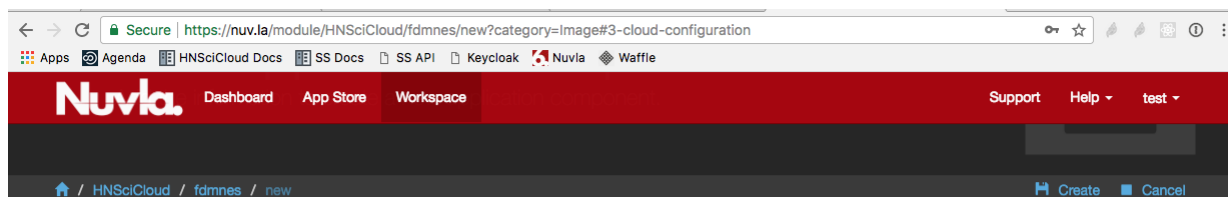
Operating System Details

Cloud Configuration

Application Parameters

Application Workflows

Authorizations



Description	Value
Number of CPUs	
Disk in GB	
Volatile extra disk in GB	
Network type	Public
RAM in GB	

Summary

Cloud Image Identifiers and Image Hierarchy

Operating System Details

Cloud Configuration

Cloud

Application Parameters

As mentioned previously, the image must use `cloud-init` for the contextualization. In addition, it must also have a

compatible version (v2.6+, not v3.x) of python installed. Some other requirements for the images can be found in a [VMware binary image](#) KnowledgeBase article. The article targets VMware, but the requirements are more general.

3.13 Create API Key/Secret

Nuvla supports the use of generated API key/secret pairs for accessing the service. Compared to other authentication methods, they provide more control over access granted to clients accessing Nuvla via the API and command line.

The API key/secret pairs have the following advantages over using other authentication mechanisms:

- Many independent pairs can be created allowing fine-grained control over programmatic access by clients.
- API key/secret pairs created with a predefined lifetime (time-to-live, TTL), disallow access after the TTL has expired.
- Long-lived clients can use API key/secret pairs with an unlimited lifetime to simplify credential management.
- **Any API key/secret can be revoked at any time and independently of any other credentials.**

The internal process for handling authentication when using API key/secrets is the following:

1. Create an API key/secret pair, saving the secret provided in the response.
2. Use the API key/secret to authenticate against the Nuvla service.
3. The server responds with a time-limited session cookie that must be used with subsequent requests to the server.
4. When the session cookie expires, the client must use the API key/secret pair to re-authenticate with the server.

While the API key/secret can be revoked, the session cookie is an **irrevocable access token** with limited lifetime. Consequently, after an API key/secret has been revoked, there is a window of time where active session cookies will still allow access. **The maximum lifetime of a session cookie is fixed at 24 hours.**

Note: Currently the browser interface does not support generation of API key/secret pairs. To work around this limitation, the API key/secret pairs can be generated from the command line.

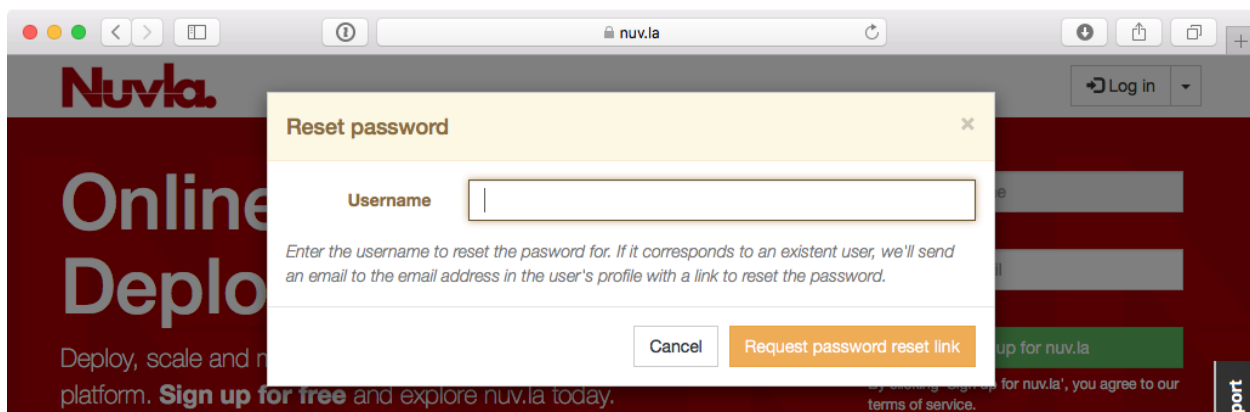
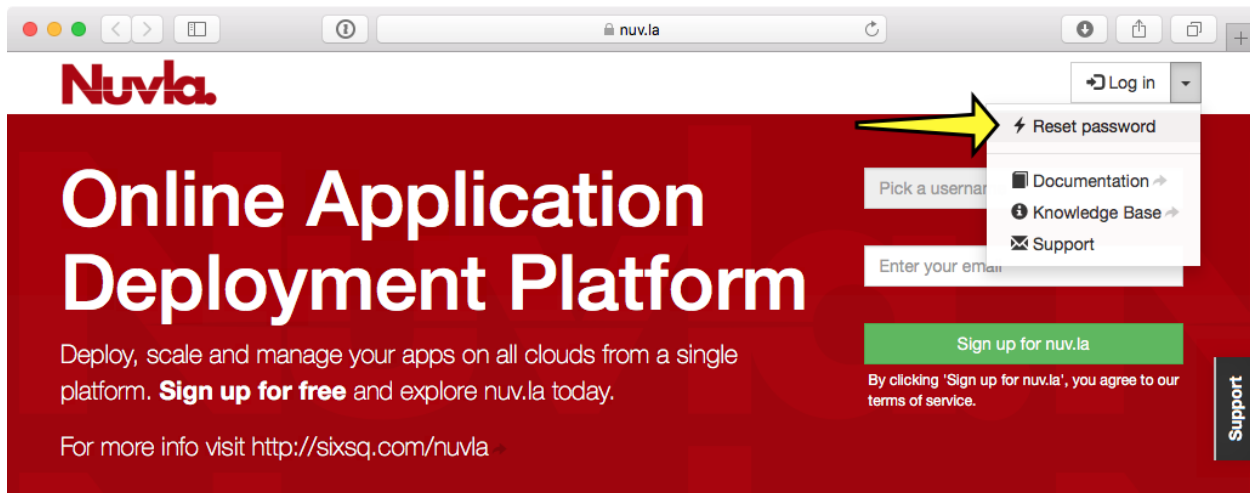
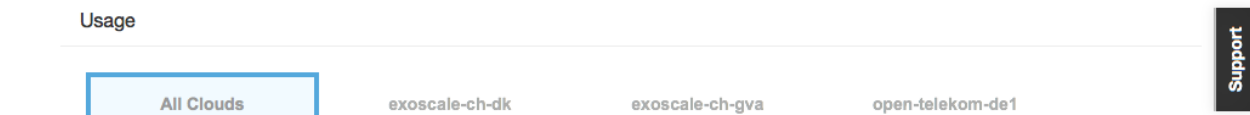
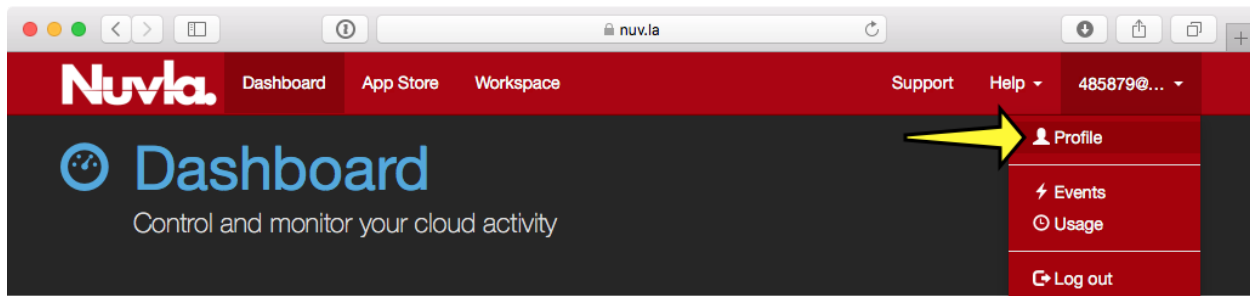
3.13.1 Enabling Account Password

For users authenticating via eduGAIN and Elixir AAI, **you must enable a password for your Nuvla account.** To do this, use the password reset function.

1. Log into Nuvla normally with your eduGAIN or Elixir AAI identity provider.
2. After logging in, capture your full username from your profile page. You will need the **complete** username.
3. Logout from Nuvla and start the password reset process. The link is shown in the screenshot below:
4. Provide your complete username in the password dialog. This will send a confirmation email to your address.
5. Visit the link provided in the email. This will then send you another email with a randomly generated password.
6. Using your username and the randomly generated password, log into the server via the command line using the *ss-curl* alias. Details on how to setup the *ss-curl* alias can be found in the [SlipStream documentation](#).

Note: Be sure to setup the *ss-curl* alias. See the SlipStream [cURL](#) documentation for setting up the correct alias.

With all that completed, you can now create an API key/secret.



3.13.2 Credential Creation

Once logged in, you can then generate new API key/secret credentials. The details can be found in the [SlipStream API Documentation](#) (API Key and Secret section).

Create a template with the information necessary to create the credential:

```
{
  "credentialTemplate" : {
    "href" : "credential-template/generate-api-key",
    "ttl" : 86400
  }
}
```

The *ttl* parameter for the API key/secret lifetime (TTL) is optional. If not provided, the credential will not expire (but can still be revoked at anytime.) The TTL value is in seconds, so the above time corresponds to 1 day. Name the file something like `create.json`.

To actually create the new credential:

```
$ ss-curl https://nuv.la/api/credential \
-X POST \
-H 'content-type: application/json' \
-d @create.json
```

```
{
  "status" : 201,
  "message" : "created credential/05797630-c1e2-488b-96cd-2e44acc8e286",
  "resource-id" : "credential/05797630-c1e2-488b-96cd-2e44acc8e286",
  "secretKey" : "..."
```

Note carefully the secret (`secretKey`) that is returned from the server. The “key” is the value of “resource-id”. This secret is not stored on the server and cannot be recovered.

3.13.3 Using the API Key/Secret

You can use the API key/secret to log in via the [REST API](#), [Python API](#), [Clojure API](#), and [Libcloud driver](#).

3.13.4 Revoking an API Key/Secret

When logged into Nuvla via the API, revoking an API key/secret corresponds to deleting the credential. This can be accomplished by doing the following:

```
$ ss-curl \
-X DELETE \
https://nuv.la/api/credential/05797630-c1e2-488b-96cd-2e44acc8e286
```

Once the credential is deleted/revoked, it can no longer be used to authenticate with Nuvla.

3.14 Benchmark

Nuvla provides a benchmarking infrastructure, which can be used by any authenticated user.

The benchmark entries themselves comply with an open schema where users can publish benchmarking data in a consistent and flexible way. The published benchmark record requires an associated Service Offer attribute, where all the instance's resources are described, and a Credentials attribute, associated with the user running/publishing the benchmark. The remaining message attributes are optional, letting users publish any sort of metrics as long as they belong to a predefined namespace (as described in the official [SlipStream API Documentation](#)).

The benchmarks can then be used to select the best performing clouds and service offers over time, continuously.

To illustrate this feature and build our own knowledge base, we publish benchmarks resulting from our continuous monitoring system. The following clouds are currently covered, and we will expand this coverage to more clouds and service offers already configured on the Nuvla service:

- Open Telekom Cloud (OTC)
- Exoscale Dietikon
- Exoscale Geneva

The published benchmarks are obtained by running the *'Unixbench'* suite to measure CPU performance through fast synthetic benchmarks like *Whetstone* and *Dhrystone*, every hours.

As an example, the following image shows the CPU performances on both Exoscale data centers over a period of 12 hours. In the image it is possible to evaluate the consistency of the CPU performance by plotting the average benchmark scores plus two edge (low and high) percentiles, which provide a general view of the CPU MIPS range. The shorter the range, the more consistent the CPU performance is.



For more details on the benchmark resource, including usage examples, refer to the [benchmark API documentation](#).

3.15 HPCaaS

High Performance Computing (HPC) involves the execution of a CPU-intensive application with a particular set of input parameters and input data sets. Because of the large CPU requirements, these applications normally use the Message Passing Interface (MPI) to create a high-performance platform from a sizable number of discrete machines.

These types of applications are naturally job or task-based and historically have been run on batch systems such as [Torque](#), [SLURM](#), or [HTCondor](#). This job-based focus, however, fundamentally conflicts with the virtual machine

focus of IaaS cloud infrastructures.

There are two approaches for running HPC applications on the HNSciCloud-RHEA hybrid cloud platform:

1. Direct use of virtual machines on the hybrid cloud platform to run a single MPI-based HPC job on a “raw cluster”.
2. Deployment of a dedicated batch system within the hybrid cloud platform to manage multiple HPC jobs.

Of the two approaches, the first is strongly preferred because it avoids the incidental complexity of managing a batch system.

3.15.1 Raw Cluster

In this case, a collection of virtual machines are provisioned and joined together via SSH to create a raw cluster, dedicated to running a single job.

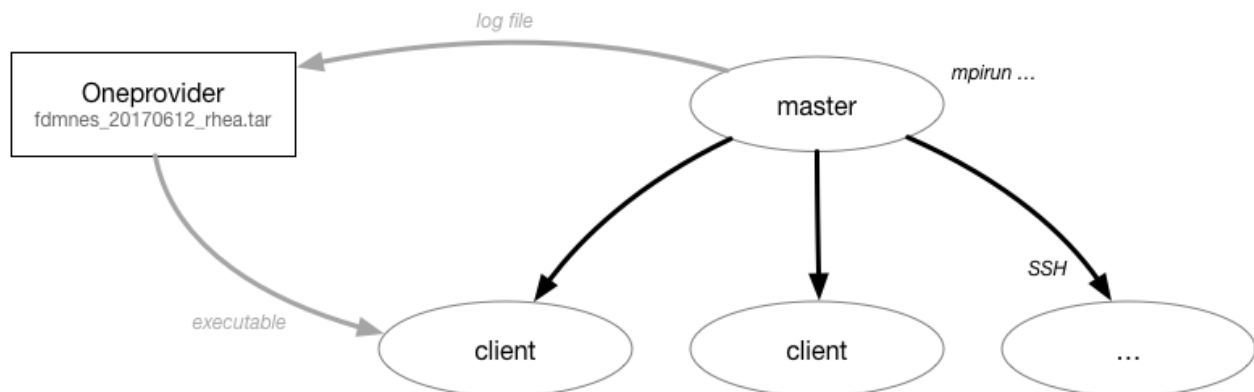
The advantages of this approach are:

- The resources can be tuned precisely to the requirements of the calculation.
- The user avoids the administrative overhead in having to deploy and manage a batch cluster.
- Questions of authentication and authorization are sidestepped because the deployed cluster serves a single user and single job.

The only apparent disadvantage is the overhead in provisioning the resources. Most cloud infrastructure can now provision all resources within a few minutes, so the overheads are already small. Taking into account that typical HPC jobs last hours or days, the overhead is miniscule in comparison.

An example application has been provided that demonstrates how to create a parameterized SlipStream application that creates and uses a raw cluster. The example uses the FDMNES application from the European Synchrotron Radiation Facility ([ESRF](#)).

The architecture of the application is shown in the figure below. There is a single master node and any number of worker nodes. The application will automatically configure all of the nodes so that the “mpiruser” account can control all of the workers nodes via SSH.



The example application will download and install a tarball containing the application to be run from a Onedata deployment. It will then start the application with the `mpirun` command, wait for the job to complete, and then provide the results in the application report, as well as pushing them to the Onedata system.

3.15.2 Batch System

If you prefer instead to use a proper batch system to handle your HPC application, then you can deploy a proper batch system on the cloud. An example SlipStream application for a [Torque cluster](#) is provided.

Warning: Note that this example is significantly out of date, but provides an example on how to create your own batch system deployment.

This example shows how to setup the batch system configuration, create users on each node of the cluster, and allow for SSH access between nodes. Your batch system may also need to setup a shared file system.

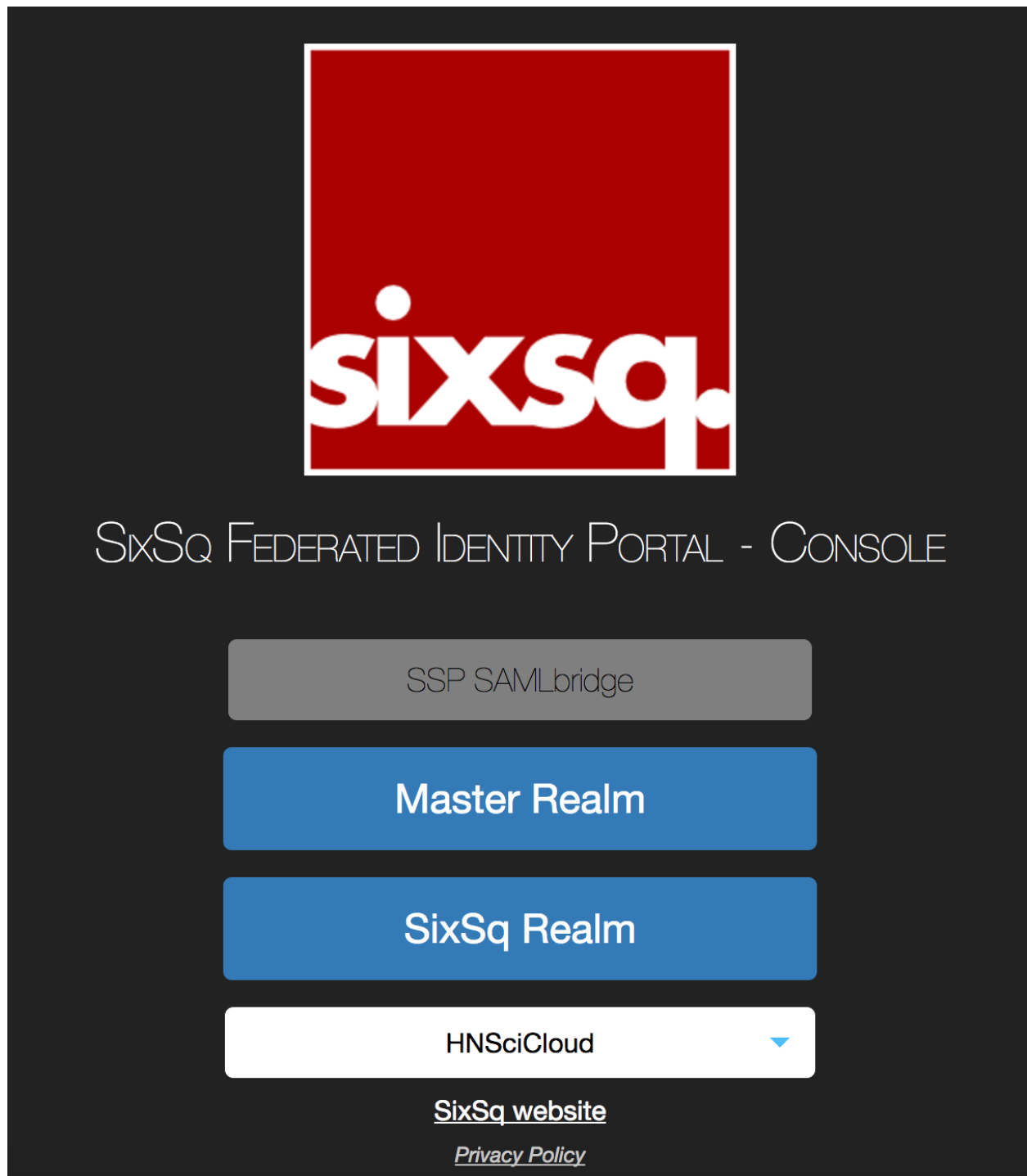
CHAPTER 4

Administrators

This section contains howtos for common tasks that will be carried out by account administrators.

These account administrators are supposed to have followed the [Account Activation](#) steps, and already have been granted the *admin* privileges to their tenant/realm in [SixSq's Federated Identity Portal](#).

To access the account management portal in Keycloak, users need to select their tenant (same as realm) in [SixSq's Federated Identity Portal](#)



and login.

4.1 Cloud Provider Configuration

Cloud accounts for each organization (tenant) have been created on the three cloud infrastructures that provide resources to the Rhea Consortium's hybrid cloud platform—Advania, Exoscale, and OTC. An organization's users share

access to these accounts via Nuvla. The organization's (tenant's) administrator decides who has access to these credentials.

4.1.1 Cloud Accounts

The general configuration of the cloud accounts follow a hierarchical approach, as shown in the diagram below.

For Exoscale, there is a top-level organization that owns and manages all the Buyers Group tenants. On each tenant then, the respective organization administrator is also given ownership.

For OTC and Advania, this top-level organization does not exist but the Buyers Group tenants are structured the same way - with the respective tenant administrator as owner and a SixSq (monitoring) account as a technical user.

With this setup, it is ensured that all the cloud accounts will be automatically setup in Nuvla, given that users have the necessary rights to provision resources.

4.1.2 Granting Access

To grant access to the shared credentials and to allow users to deploy applications on the clouds, each account manager should:

1. Login to [SixSq's Federated Identity Portal](#)
2. Select the users (or groups of users) who need provisioning access, and assign them with the role **can_deploy** (which has already been created).

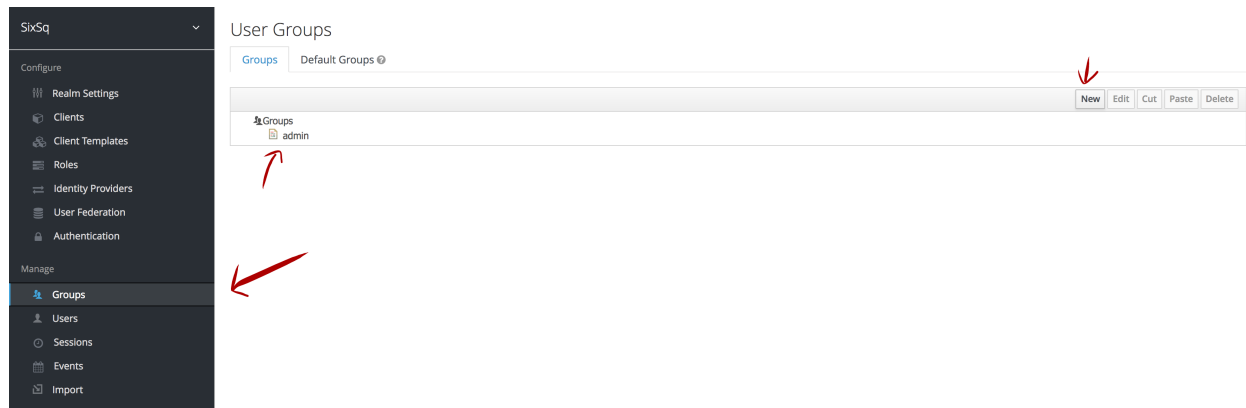
Once this is done, the affected users will automatically get access to the cloud credentials for Exoscale, OTC and Advania in Nuvla. The assignment of groups or roles is done during the login process, so users may have to logout and login again to have access to new groups or roles.

4.2 Manage Groups

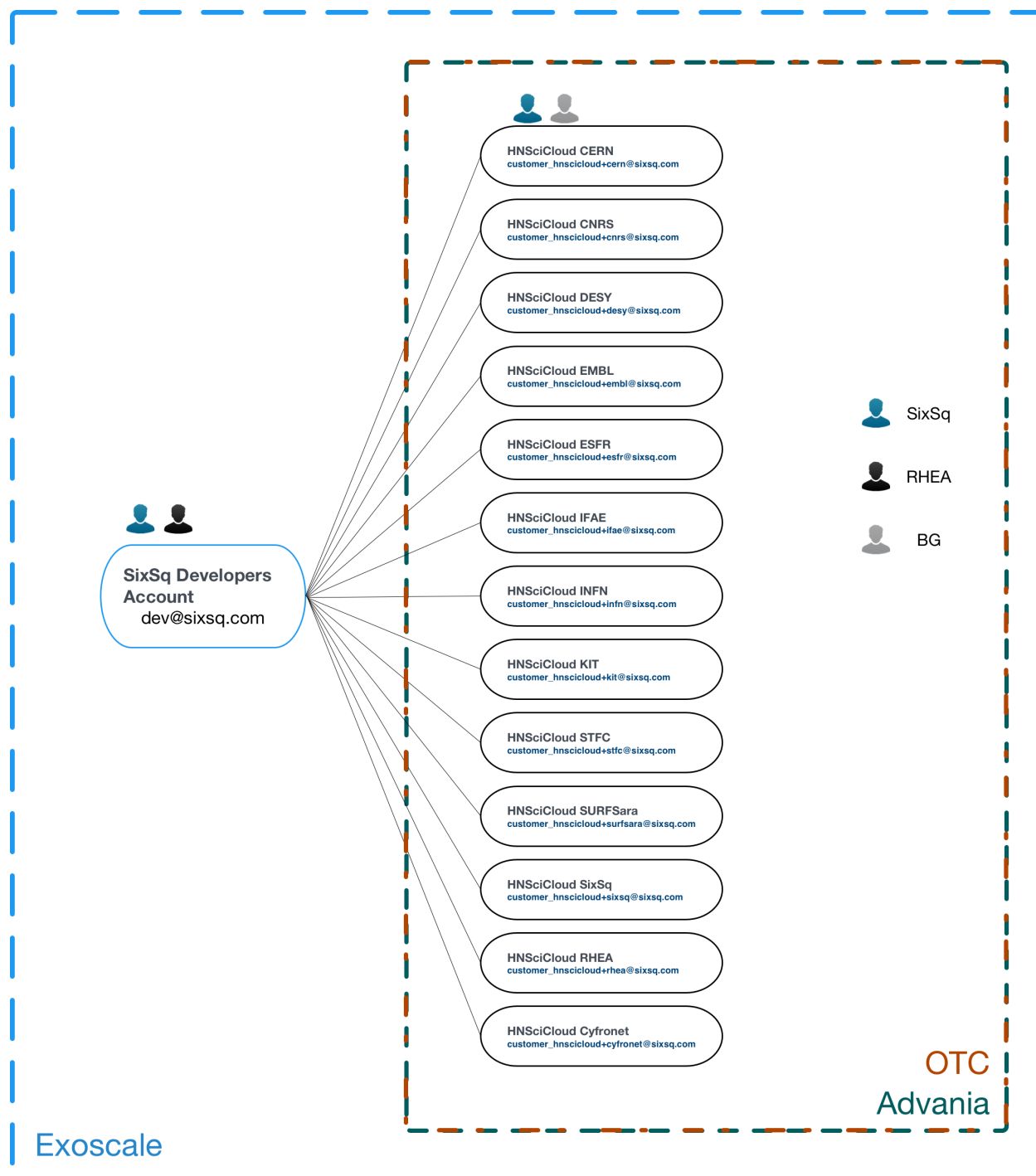
To simplify, groups are to be considered the same as sub-tenants, as they provide the required isolation level.

Groups can be managed through the Keycloak's UI within [SixSq's Federated Identity Portal](#).

Upon registration in Nuvla, account managers will be given (if requested) *admin* privileges in their respective tenant. Tenant account managers belong to the **admin** group in Keycloak, and have full management privileges within that realm.



Below follows a comprehensive list of actions that can be performed group-wise.



The screenshot shows the SixSq user management interface. On the left is a sidebar with a 'Configure' section containing 'Realm Settings', 'Clients', 'Client Templates', 'Roles', 'Identity Providers', 'User Federation', and 'Authentication'. Below this is a 'Manage' section with 'Groups', 'Users' (highlighted), 'Sessions', 'Events', and 'Import'. The main content area is titled 'Users' and shows a breadcrumb '9ad67dc3aded28dac61e0f5b78b76109acd69c07@elixir-europe.org'. Below this is a tabbed interface with 'Details', 'Attributes', 'Credentials', 'Role Mappings' (active), 'Groups', 'Consents', 'Sessions', and 'Identity Provider Links'. The 'Role Mappings' tab shows four columns: 'Realm Roles', 'Available Roles' (with a dropdown menu showing 'can_deploy', 'data-manager', 'one-data-special-role', and 'personalRole'), 'Assigned Roles' (with 'offline_access' and 'uma_authorization'), and 'Effective Roles' (with 'offline_access', 'uma_authorization', and 'view_accounting'). There is an 'Add selected >' button under 'Available Roles' and a '<< Remove selected' button under 'Assigned Roles'. At the bottom, there is a 'Client Roles' section with a dropdown menu and the text 'Select client to view roles for client'.

4.2.1 Create New Group

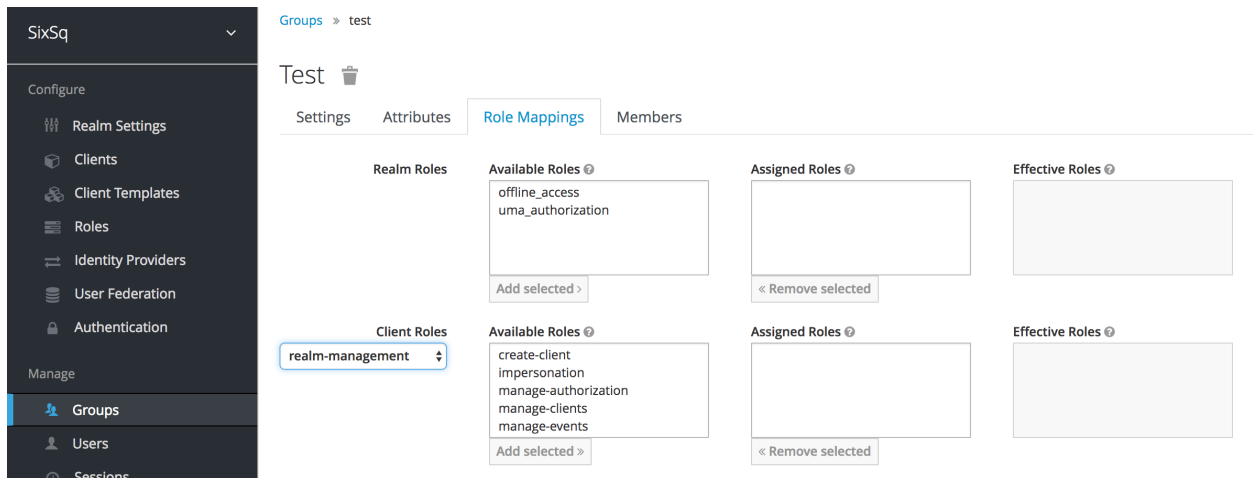
Groups can be nested within each other which allows account managers to define hierarchies. Here we refer to **top level group** and **subgroup** as being the main group and the ones within it, respectively.

Top Level Group

Without selecting any group (as shown on the figure above), click *New* and enter your group name.

The screenshot shows the 'Create group' form in the SixSq interface. The sidebar is the same as in the previous screenshot. The main content area is titled 'Create group'. Below the title is a 'Name' field with a red asterisk, containing the text 'test'. The field is highlighted with a red oval. Below the field are 'Save' and 'Cancel' buttons.

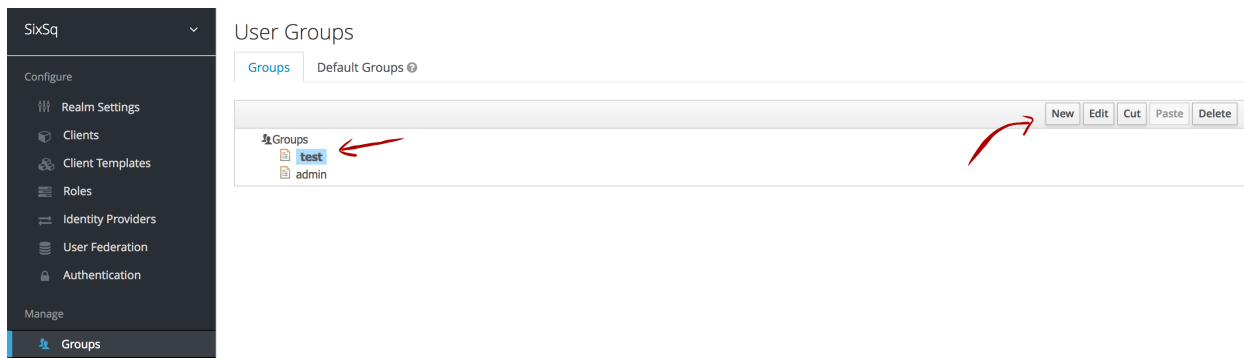
Upon creation, a new page will show up with the group settings, as shown below:



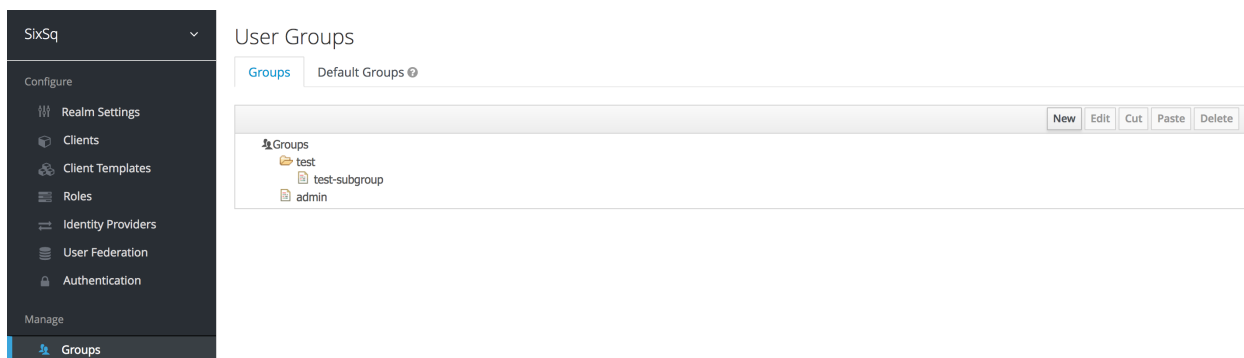
Here is where the account managers can rename the group, assign custom attributes, see the group members (empty by default), and map the group roles.

Subgroup

To create a group that belongs to a top level one, the creation process is exactly the same as above, but the account manager **has** to select the respective top level group before clicking on *New*:



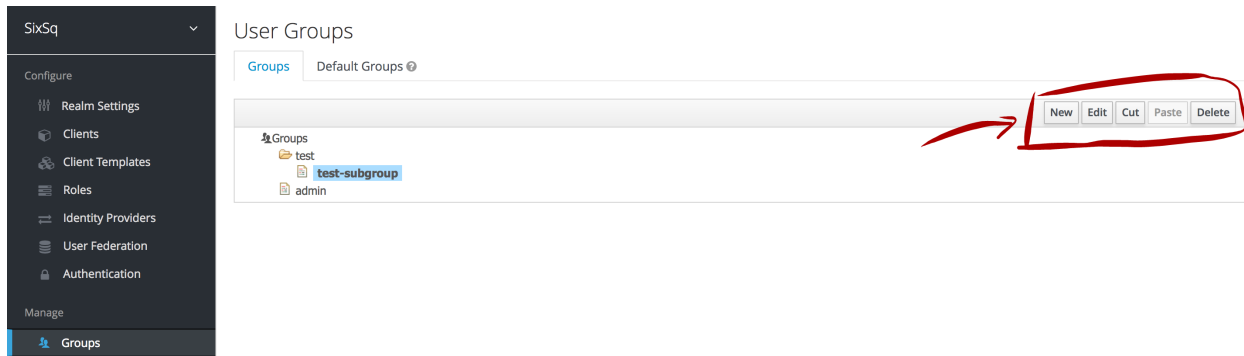
Once created:



NOTE: please note that subgroups will by default inherit the role mappings from the parent top level groups

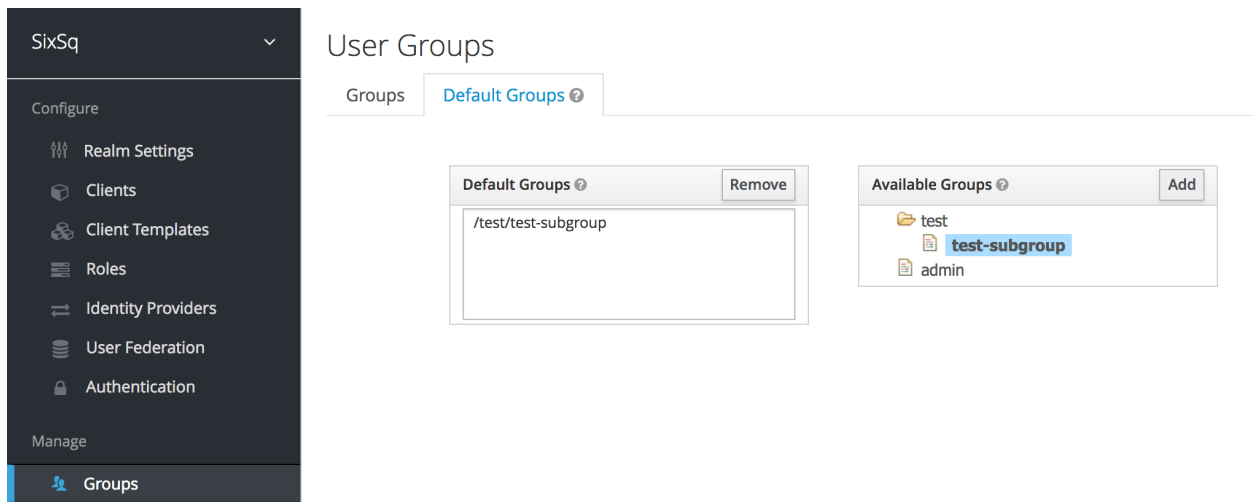
4.2.2 Edit, Cut, Paste and Delete Groups

The button panel on the right corner of the groups' interface allows all the basic group operations:



4.2.3 Default Groups

“Set of groups that new users will automatically join”



As the above figure shows, new users will now automatically be assigned to the subgroup *test-subgroup*.

4.3 Managing Roles

Roles can be managed through the Keycloak's UI within SixSq's [Federated Identity Portal](#).

There are different type of roles and actions that can be performed to manage them:

4.3.1 Realm Roles

Roles that are applied on the tenant level.



Roles

Realm Roles Default Roles

Search...

Role Name	Composite	Description	Actions
offline_access	False	\${role_offline-access}	Edit Delete
uma_authorization	False	\${role_uma_authorization}	Edit Delete

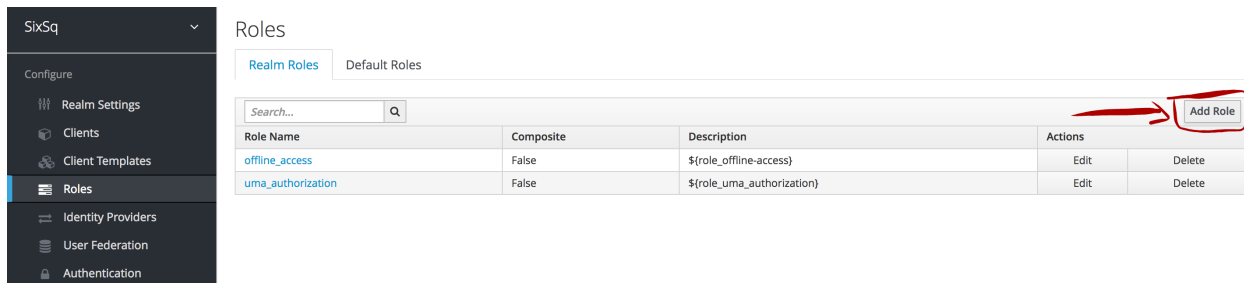
Add Role

Default Roles

If selected, default roles will automatically be assigned to new users.

Create Roles

To create a new role, simply click on *Add Role*:



Roles

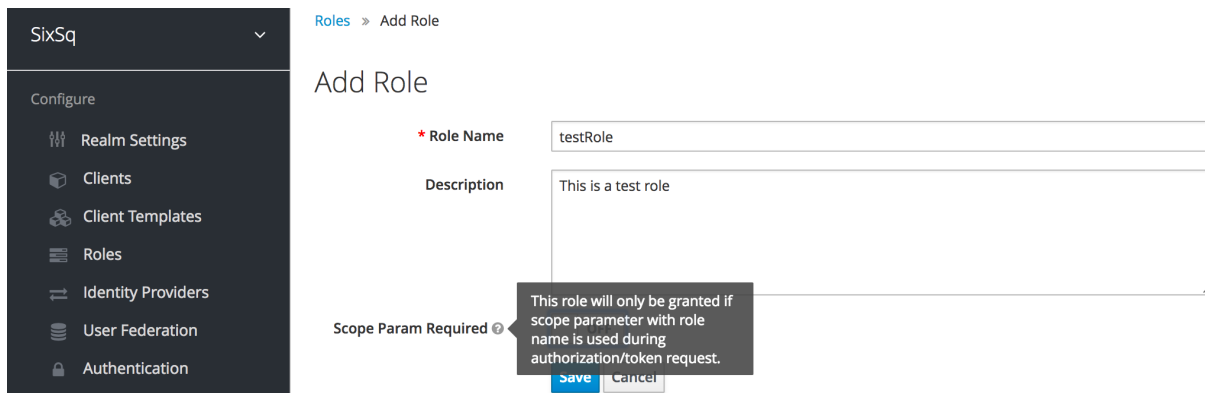
Realm Roles Default Roles

Search...

Role Name	Composite	Description	Actions
offline_access	False	\${role_offline-access}	Edit Delete
uma_authorization	False	\${role_uma_authorization}	Edit Delete

Add Role

This will open a new form where the account manager can define the role's name, description and whether the role will only be granted if scope parameter with role name is used during authentication/token request:



Roles » Add Role

Add Role

* Role Name: testRole

Description: This is a test role

Scope Param Required: ☐

This role will only be granted if scope parameter with role name is used during authentication/token request.

Save Cancel

Once created, account managers will then also have the option to assign composite roles:

SixSq
Configure
Realm Settings
Clients
Client Templates
Roles
Identity Providers
User Federation
Authentication
Manage
Groups
Users
Sessions
Events
Import

Roles » testRole
TestRole
Role Name: testRole
Description: This is a test role
Scope Param Required: OFF
Composite Roles: When this role is (un)assigned to a user any role associated with it will be (un)assigned implicitly.
Save Cancel
Composite Roles
Realm Roles
Available Roles: offline_access, uma_authorization
Add selected »
Associated Roles: « Remove selected
Client Roles: Select client to view roles for client

NOTE: by default, new rules do not become “Default Roles” for that realm.

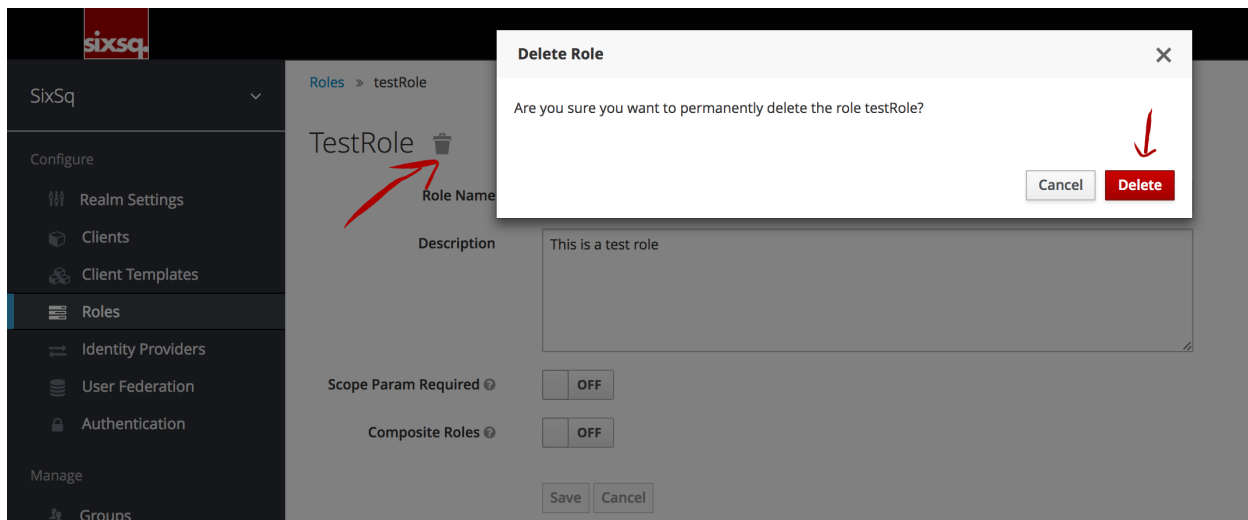
Edit and Delete Roles

To edit, simply click on the role name (from the list of roles). To delete, once inside the role edition page, click on the bin icon next to the role name:

SixSq
Configure
Realm Settings
Clients
Client Templates
Roles
Identity Providers
User Federation
Authentication

Roles
Realm Roles
Default Roles
Search...
Add Role

Role Name	Composite	Description	Actions	
offline_access	False	\${role_offline-access}	Edit	Delete
testRole	False	This is a test role	Edit	Delete
uma_authorization	False	\${role_uma_authorization}	Edit	Delete



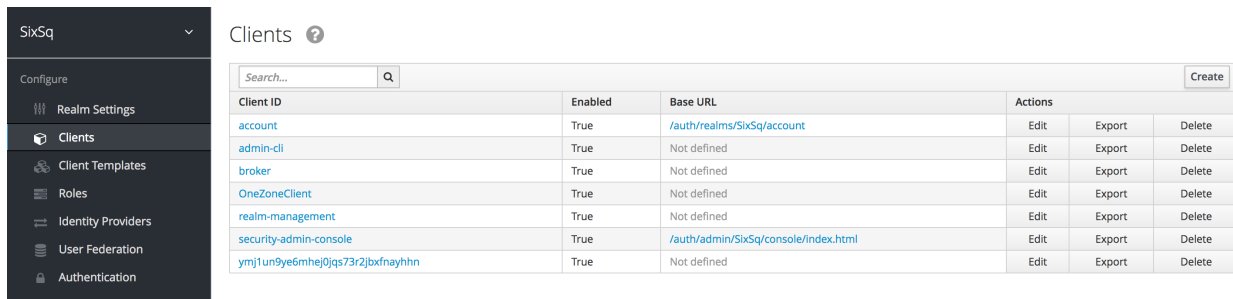
4.3.2 Client Roles

Keycloak clients are trusted browser apps and web services in a realm. These clients can request a login.

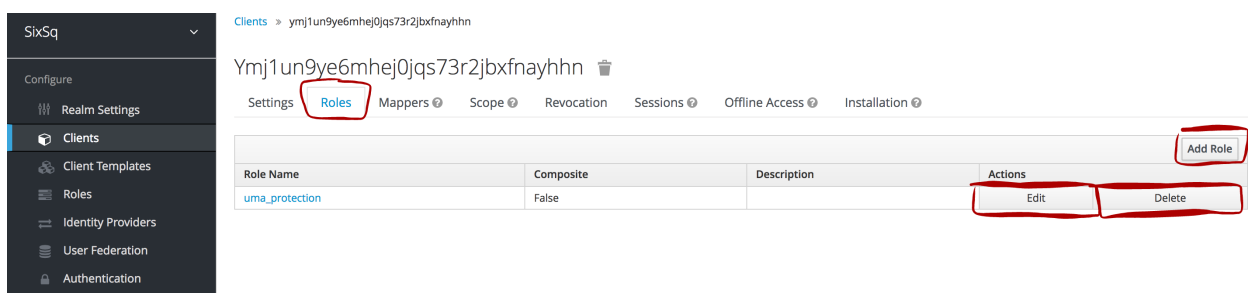
Account managers can also define client specific roles. IT IS NOT RECOMMENDED that account managers change the roles of already existing clients, as the default tenant clients (and respective Client Templates) are not configured to propagate the user client roles (which are defined on the *Clients* section under the *Scope* tab, for each client).

Manage Client Roles

To manage client roles, account manager should first select the desired client from the list



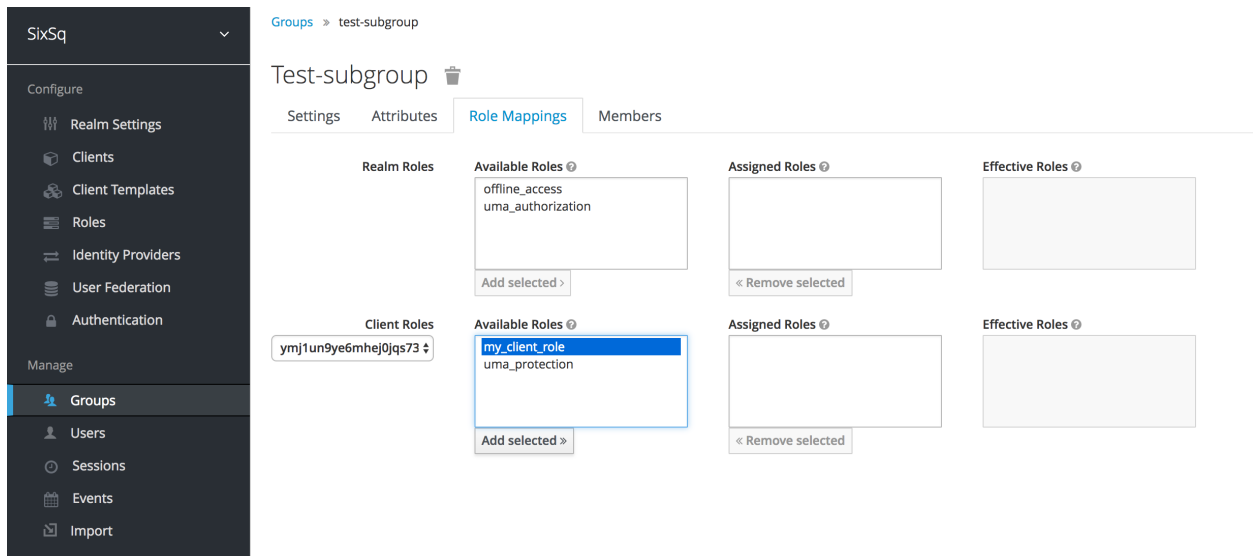
and then click on the *Roles* tab. Here, the account managers will get a list of the client roles and the chance to add new ones as well as edit and delete existing ones:



To add and modify client roles, the interface is exactly the same as stated above for Realm roles.

4.3.3 Mapping Realm and Client Roles to Groups

The instructions on how to map a role to a group can be found in [here](#). Once in the group page, switch to the “Role Mappings” tab and select the desired roles, as shown below.



4.4 Quota

Nuvla protects users from overuse by applying a quota mechanism. The same feature gives administrators a powerful tool to control usage on the hybrid-cloud platform. The feature takes the form of an extended quota resource, which provides fine grain control over who is allowed to consume how many resources, of which type and on which cloud.

When deploying new resources, Nuvla checks if the requested deployment will result in the user exceeding its quota. Only if this is not the case, will the deployment be authorised. Administrators can set quotas, including:

- CPU
- RAM
- Disk

Beyond these simple limits, the quota feature offers great flexibility, such as defining limits based on aggregations such as *count*, *sum* and *max* on any attributes of a virtual machine.

The administrator can define a range of ACLs for each quota. The Nuvla monitoring engine tracks usage and compares this usage with the most applicable quota. This therefore allows the provisioning engine to ensure that users do not exceed their quota.

Since several quotas can be concurrently applicable at any time (e.g. user, group or even general tenant level), the most appropriate quota available will be applied when assessing if a user has or is about to exceeded its quota or not.

To change quotas, the organisation (tenant), the group (sub-tenant) owners, or the user must place a request with Nuvla [Support](#) desk. Future releases will allow owners to change quotas themselves, within the limits set by administrators.

More details on how to use this feature can be found in the [monitoring cloud activities](#) API documentation.

4.5 Account Management API

It is also possible to manage accounts in Keycloak through its REST API: <http://www.keycloak.org/docs-api/3.1/rest-api/index.html>

4.6 Blacklisting Users

One of the desired user management features for the HNSciCloud project is the ability to block users. This is an action that can be carried by account managers, at any time, removing the user's access to all client applications in the respective tenant.

There are several ways an account manager can reduce the privileges or even block a user:

4.6.1 Deny User Access to Specific Resources

In some cases, the account manager might only want to grant or remove privileges from a certain user, which will later translate on a forbidden access to a specific resource.

Assuming that all the ACLs have been defined, the account manager should use the roles and groups available in Keycloak to manage the users' access to the deployed resources.

These actions are documented in the [Manage Groups](#) and [Managing Roles](#) sections.

4.6.2 Blocking a Local User

On the occasional scenario where there are local user accounts setup through Keycloak, the account manager can block these users through 3 different ways:

Disable the account

Users with disabled accounts cannot login. The account manager should go the user's view and toggle the *User Enabled* button:

SixSq

Configure

- Realm Settings
- Clients
- Client Templates
- Roles
- Identity Providers
- User Federation
- Authentication

Manage

- Groups
- Users**
- Sessions
- Events
- Import

Users > test

Test

Details
Attributes
Credentials
Role Mappings
Groups
Consents
Sessions
Identity Provider Links

ID: 32e7d658-3e9a-4dfb-a677-ad1117d6a106
Created At: 6/29/17 11:56:19 AM
Username: test
Email:
First Name:
Last Name:
User Enabled: ☐ OFF
Email Verified: ☐ OFF
Required User Actions:
Locale:
Impersonate user:

Change the password

An obvious way to block a user is to simply disable his/her credentials or even change the password:

SixSq

Configure

- Realm Settings
- Clients
- Client Templates
- Roles
- Identity Providers
- User Federation
- Authentication

Manage

- Groups
- Users**
- Sessions
- Events
- Import

Users > test

Test

Details
Attributes
Credentials
Role Mappings
Groups
Consents
Sessions
Identity Provider Links

Manage Password

New Password:
Password Confirmation:
Temporary: ☒ ON

Disable Credentials

Disableable Types:
Disable Credential Types:

Remove the user

Finally, a definite and also obvious way to block a user, is simply delete the account:

Authentication	32e7d658-3e9a-4dfb-a67...	test				Edit	Impersonate	Delete
Manage	3e0447e3-c283-4302-914...	test10				Edit	Impersonate	Delete
	10417582-57a3-4bd4-932...	test11				Edit	Impersonate	Delete

4.6. Blacklisting Users

47

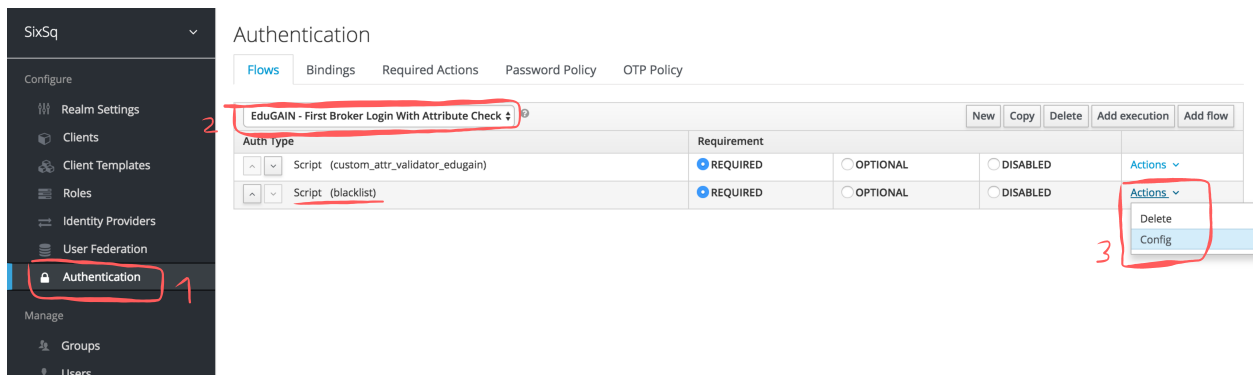
4.6.3 Blocking an External User

In most cases, users will be registering into Keycloak (and consequently into the corresponding client applications and services) through external IdPs. These kind of accounts do not have an associated password and therefore it is recommended to have a different approach when blocking users.

To block a user from an external IdP, account manager should add their unique usernames into a **BLACKLISTED_USERNAMES** list, which is crosschecked against every user during login. If a user is blacklisted, he'll be able to register into Keycloak but his/her login will be interrupted and a "blacklisted" message will be shown.

To blacklist a user, account manager should first find the respective username from the users list. Let's assume we want to block user *foo@bar.comhttp://myidp.com/login!etc.* Account managers should then do:

1. go to *Authentication*
2. select in which federation should the user be blacklisted
3. click on *Config* from the *blacklist* script.



This will open a script where account managers can then find a global variable (a list) called *BLACKLISTED_USERNAMES*. The username to block shall then be put into that list, as a string, like shown below:

Authentication Flows » EduGAIN - First Broker Login With Attribute Check » blacklist

Blacklist

ID: b1bb3005-0af6-488b-8995-36ad76fafc2e

Alias: blacklist

Script Name: blacklist

Script Description: blacklist

Script Source:

```

1  /* to blacklist users, add their usernames to the following list. ex:
2     BLACKLISTED_USERNAMES = ['usernameAB', 'user123']
3  BLACKLISTED_USERNAMES = ['foo@bar.comhttp://myidp.com/login!etc']
4
5
6  /*
7   * Template for JavaScript based authenticator's.
8   * See org.keycloak.authentication.authenticators.brower
9   * .ScriptBasedAuthenticatorFactory
10  */
11
12  // import enum for error lookup
13  AuthenticationFlowError = Java.type("org.keycloak.authentication
14  .AuthenticationFlowError");
15  AuthenticationFlowError =
16  Java.type("org.keycloak.authentication.AuthenticationFlowError");
17  Response = Java.type("javax.ws.rs.core.Response");
18  MediaType = Java.type("javax.ws.rs.core.MediaType");
19
20  /**
21   * An example authenticate function.
22   *
23   * The following variables are available for convenience:
24   * user - current user {@see org.keycloak.models.UserModel}
25   * realm - current realm {@see org.keycloak.models.RealmModel}
26   * session - current KeycloakSession {@see org.keycloak.models
27   * .KeycloakSession}
28   * httpRequest - current HttpRequest {@see org.jboss.resteasy.spi
29   * .HttpRequest}
30   * script - current script {@see org.keycloak.models.ScriptModel}
31   * clientSession - current client session {@see org.keycloak.models
32   * .ClientSessionModel}
33   * LOG - current logger {@see org.jboss.logging.Logger}
34   *
35   * You one can extract current http request headers via:
36   * httpRequest.getHeader() or httpRequest.getHeader("Content-Type")

```

Save Cancel

Then just click **Save**, and that's it. For any other users to be blocked, the procedure is the same, just appending their username to the list above.

Warning: Blocking a user does not revoke nor destroy his/her current session. Blocking actions only affect users on their next login.

4.7 Whitelisting Users

The ability to only permit login to a predefined set of users is also possible available in Keycloak.

As for **blacklisting**, the account manager can also define a list of usernames that are allowed to login through IdPs into Keycloak and respective clients.

By default new users are not entitled with anything else than the roles to manage and view their own user account, but obviously this can be changed by the account manager by either setting default groups and roles or assigning these to users individually.

4.7.1 Whitelisting a Local User

Local user registration is not enabled in the current SixSq Fed-ID portal, thus new local users have to be manually added by the account manager by creating new users through the “Users” section in the Keycloak UI.

Users

4.7.2 Allowing an External User

By default, whitelisting for external users is disabled.

In cases where the account manager wants to restrict the login process to a set of known users coming from either eduGAIN or ELIXIR, he/she needs to enable the whitelisting capability in the authentication flows in Keycloak:

PLEASE NOTE: blacklisting has priority over whitelisting, so if a user is both whitelisted and blacklisted, he/she will not be allowed to login and be considered blacklisted anyway.

Once enabled, account managers can add users to their whitelist in a similar way as it is done for the blacklist. Let's assume we want to add user `foo@bar.com` `http://myidp.com/login!etc` to the whitelist:

1. go to *Authentication*
2. select in which federation should the user be blacklisted
3. click on *Config* from the *whitelist* script.

Authentication

This will open a script where account managers can then find a global variable (a list) called `WHITELISTED_USERNAMES`. The username to allow shall then be put into that list, as a string, like shown below:

Authentication Flows » EduGAIN - First Broker Login With Attribute Check » whitelist

Whitelist

ID	8f697958-59f1-44b7-ad22-10c964ca60e2
Alias ?	whitelist
Script Name ?	whitelist
Script Description ?	List of users who are allowed to login
Script Source ?	<pre> 1 /* to whitelist users, add their usernames to the following list. ex: 2 WHITELISTED_USERNAMES = ['usernameAB', 'user123'] 3 */ 4 WHITELISTED_USERNAMES = ['foo@bar.comhttp://myidp.com/login!etc'] 5 6 /* 7 * Template for JavaScript based authenticator's. 8 * See org.keycloak.authentication.authenticators.browser 9 * .ScriptBasedAuthenticatorFactory 10 11 // import enum for error lookup 12 AuthenticationFlowError = Java.type("org.keycloak.authentication 13 .AuthenticationFlowError"); 14 AuthenticationFlowError = 15 Java.type("org.keycloak.authentication.AuthenticationFlowError"); 16 17 Response = Java.type("javax.ws.rs.core.Response"); 18 MediaType = Java.type("javax.ws.rs.core.MediaType"); 19 20 /** 21 * An example authenticate function. 22 * The following variables are available for convenience: 23 * user - current user {@see org.keycloak.models.UserModel} 24 * realm - current realm {@see org.keycloak.models.RealmModel} </pre>

Then just click **Save**, and that's it. That user can now login, provided the username is not blacklisted.

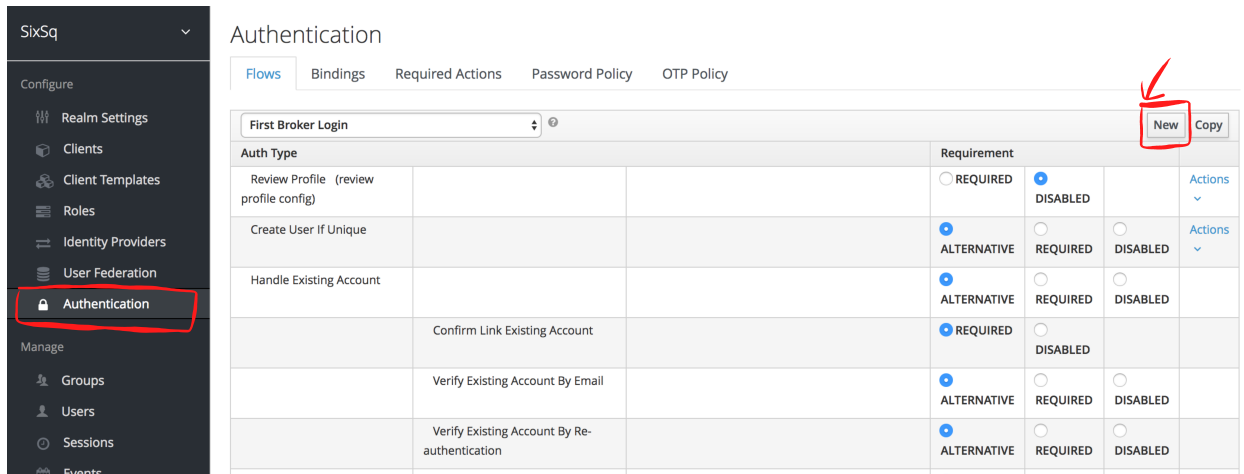
4.8 Blocking New User Logins

A useful user management feature is the ability to freeze the authentication process for one's tenant, making it impossible for new users to sign in into Nuvla.

This is a different process from blacklisting users, as this will completely block any unregistered users from signing in, while letting existing users authenticate normally.

Each account manager can achieve this within his/her own tenant by following these steps:

1. on the *Authentication* panel of the Keycloak portal, create a new Authentication Flow

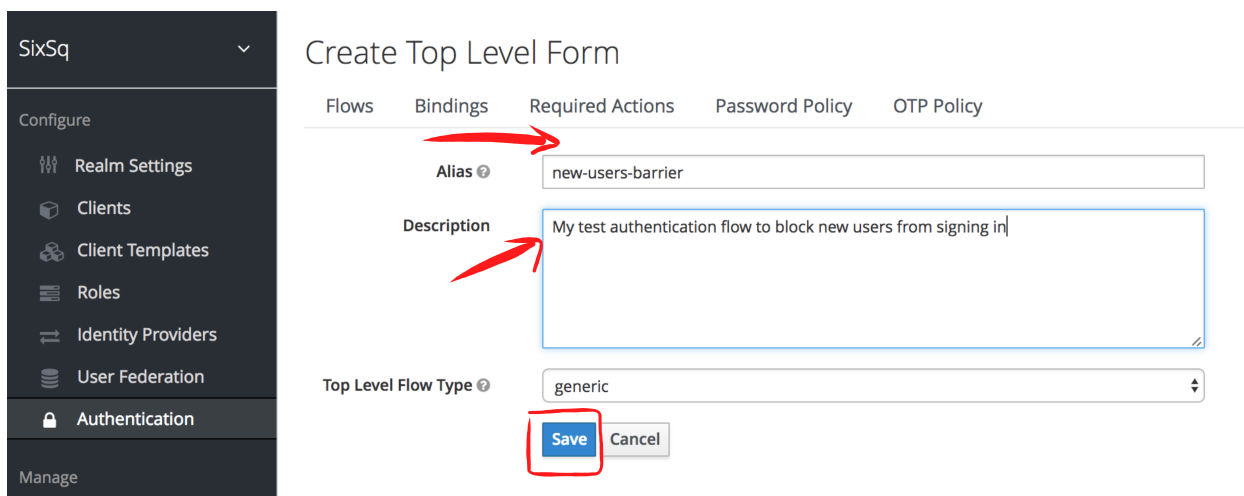


Authentication

Flows Bindings Required Actions Password Policy OTP Policy

First Broker Login

Auth Type	Requirement	Requirement	Requirement	Actions
Review Profile (review profile config)	<input type="radio"/> REQUIRED	<input checked="" type="radio"/> DISABLED		Actions
Create User If Unique	<input checked="" type="radio"/> ALTERNATIVE	<input type="radio"/> REQUIRED	<input type="radio"/> DISABLED	Actions
Handle Existing Account	<input checked="" type="radio"/> ALTERNATIVE	<input type="radio"/> REQUIRED	<input type="radio"/> DISABLED	
Confirm Link Existing Account	<input checked="" type="radio"/> REQUIRED	<input type="radio"/> DISABLED		
Verify Existing Account By Email	<input checked="" type="radio"/> ALTERNATIVE	<input type="radio"/> REQUIRED	<input type="radio"/> DISABLED	
Verify Existing Account By Re-authentication	<input checked="" type="radio"/> ALTERNATIVE	<input type="radio"/> REQUIRED	<input type="radio"/> DISABLED	



Create Top Level Form

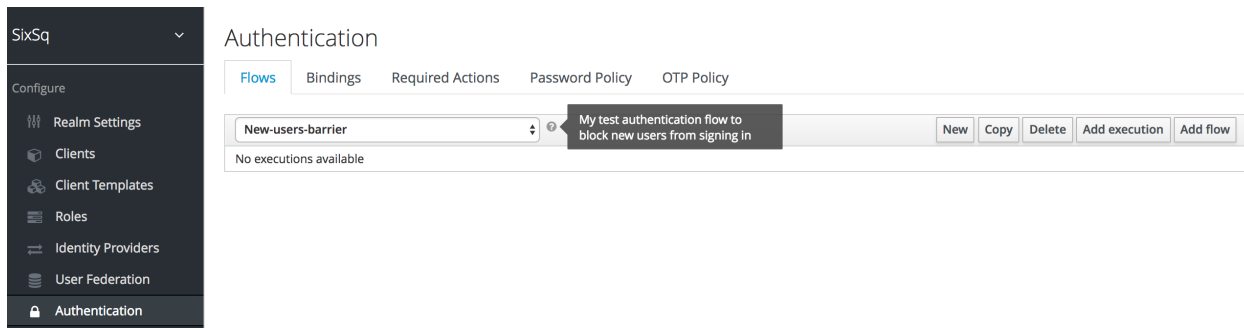
Flows Bindings Required Actions Password Policy OTP Policy

Alias new-users-barrier

Description My test authentication flow to block new users from signing in

Top Level Flow Type generic

Save Cancel



Authentication

Flows Bindings Required Actions Password Policy OTP Policy

New-users-barrier

My test authentication flow to block new users from signing in

No executions available

New Copy Delete Add execution Add flow

- do not add any other executions or flows under this new flow. Leaving it empty will basically tell Keycloak not to do anything when summoned
- since for HNSciCloud user are signing in from external Identity Providers, go to the *Identity Providers* panel in Keycloak and chose one from the list

Identity Providers

Name	Provider	Enabled	Hidden	Link only	GUI order	Actions
ELIXIR	saml	True	False	False	2	Edit Delete
eduGAIN	saml	True	False	False	1	Edit Delete

4. for the *First Login Flow* configuration parameter, select the new Authentication Flow you've created above

Identity Providers » ELIXIR

ELIXIR

Settings Mappers Export

Redirect URI

* Alias

Display Name

Enabled ☒

Store Tokens ☐

Stored Tokens Readable ☐

Trust Email ☒

Account Linking Only ☐

Hide on Login Page ☐

GUI order

First Login Flow

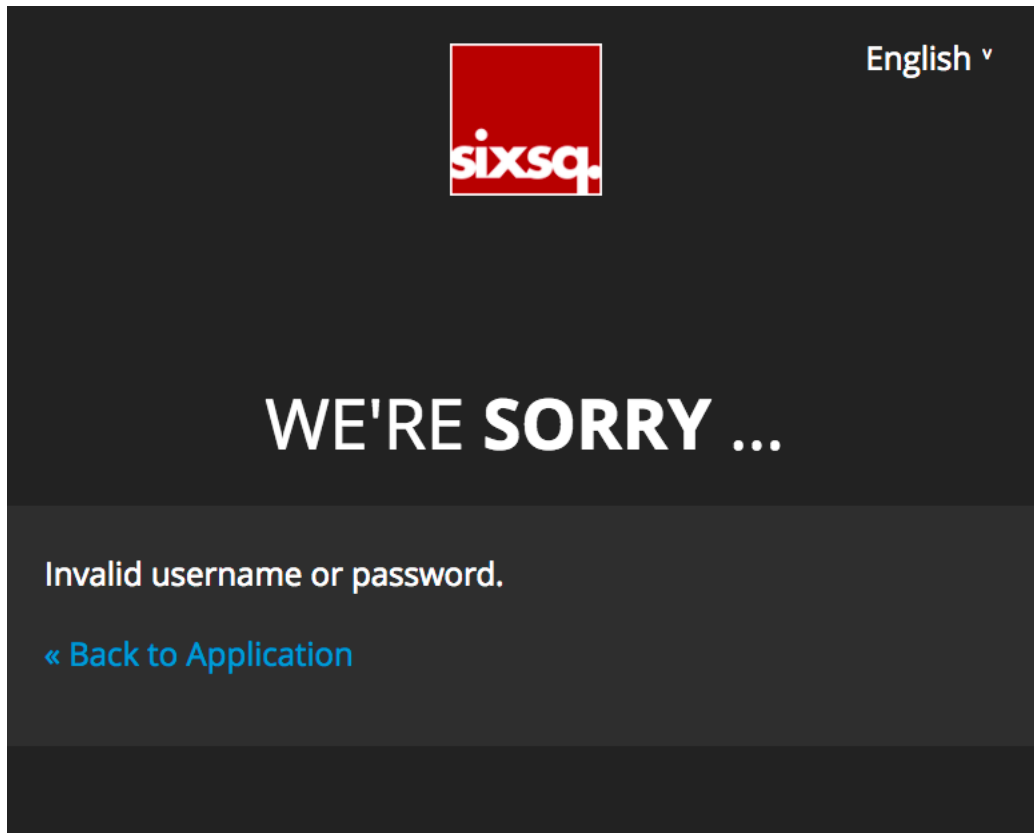
Post Login Flow

> SAML Config

5. repeat steps 3 and 4 for all the Identity Providers in the tenant

And that's it!

From this point on, new user signing in to [Nuvla](#) (or even to Keycloak directly) will get an “Invalid” login (see below) and will not be registered.



To stop this blockage, just revert the First Login Flow to what it was before, re-doing steps 3 and 4 above, but this time moving the *First Login Flow* back to **first broker login**.

This section contains howtos for common tasks that will be carried out by data coordinators.

See also:

A complete step by step [online video](#) is available, showing how to deploy the different services required to build a complete data management solution, based on Onedata.

5.1 Deploying Data Management Services

Data management services are based on Onedata technology. For general overview of Onedata and its core concepts including zones, providers and spaces please refer to the [official documentation](#).

5.1.1 Onezone

Deploying Onezone via Nuvla

Data managers are provided a possibility to automatically deploy Onezone service for their Buyer Group via Nuvla. The Onezone component definition can be found under <https://nuv.la/module/HNSciCloud/onedata/onezone>. To deploy Onezone component one needs to provide a number of input parameters to properly configure the service. The required parameters are related to the connection of the Onezone with the project's Federated Identity Provider (FedIdP) service located at <https://fed-id.nuv.la>. It is assumed that the data manager has access to the Buyer's Group realm in the FedIdP service (see [here](#)) to obtain values for the following required Onezone deployment parameters:

Input parameters More ▼

Input <code>hn-fedid-client-id</code>	<input type="text" value="OneZoneClient"/>	?
Input <code>hn-fedid-token</code>	<input type="text"/>	?
Input <code>hn-fedid-url</code>	<input type="text" value="https://fed-id.nuv.la"/>	?
Input <code>hn-tenant-name</code>	<input type="text"/>	?
Input <code>onezone-version</code>	<input type="text"/>	?

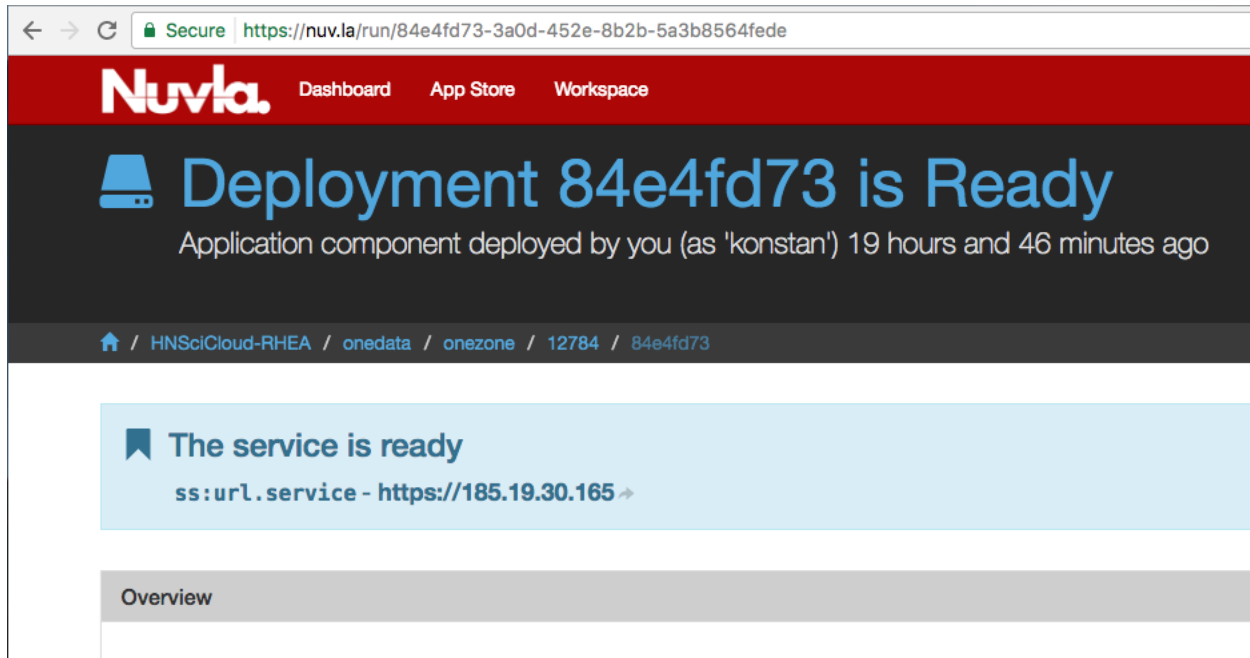
- **hn-tenant-name** - Buyer tenant name, which should correspond to the realm name in FedIdP service;
- **hn-fedid-token** - client registration token to register the deployed Onezone instance with FedIdP service. The token should be generated by the data manager in FedIdP service right before attempting the deployment. This should be done the following way:
 1. Login to your realm on <https://fed-id.nuv.la>.
 2. Go to Realm Settings -> Client Registration -> Initial Access Tokens.

The screenshot shows the SixSq admin console interface. The left sidebar contains navigation links for 'Configure' (Realm Settings, Clients, Client Templates, Roles, Identity Providers, User Federation, Authentication) and 'Manage' (Groups, Users, Sessions, Events, Import). The main content area is titled 'SixSq' and shows tabs for 'General', 'Login', 'Keys', 'Email', 'Themes', 'Cache', 'Tokens', 'Client Registration', and 'Security Defenses'. The 'Client Registration' tab is active, showing 'Initial Access Tokens' and 'Client Registration Policies'. A table lists generated tokens with columns for ID, Created, Expires, Count, Remaining Count, and Actions. One token is listed with ID '29504efc-ebc9-406d-837f-ec44fc64a12c', created on 6/26/17 at 1:36:54 PM, and a count of 1. A 'Create' button is visible in the top right of the table area.

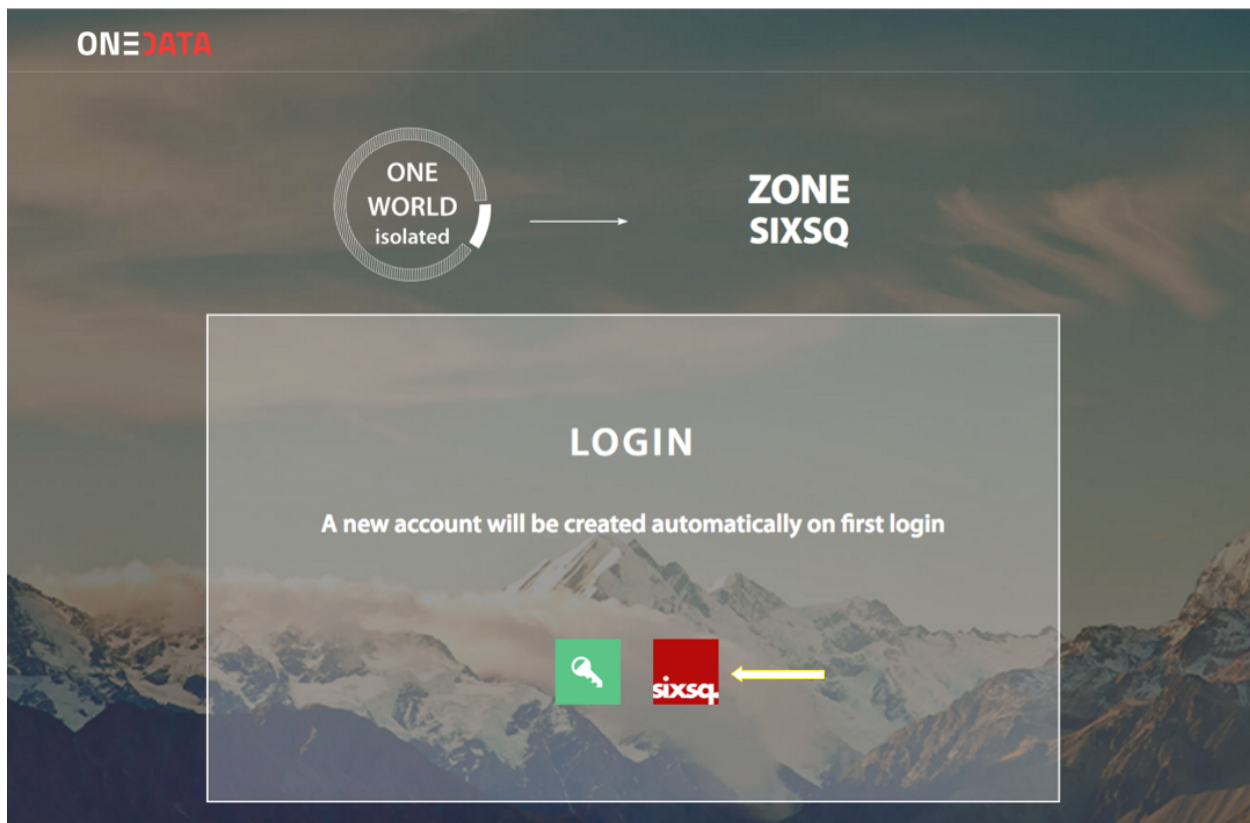
ID	Created	Expires	Count	Remaining Count	Actions
29504efc-ebc9-406d-837f-ec44fc64a12c	6/26/17 1:36:54 PM		1	1	Delete

3. Click on *Create*.
4. Select defaults and click *Save*.
5. Copy the generated token.
6. Go to onezone deployment dialog and paste the token to **hn-fedid-token** edit field.
 - **onezone-version** - the default production version will be used if the value is not provided.

When parameters are set, click on *Deploy Application Component* button. After the successful deployment of Onezone, click on the URL defined by `ss:url.service` and you will be redirected to the running Onezone service.



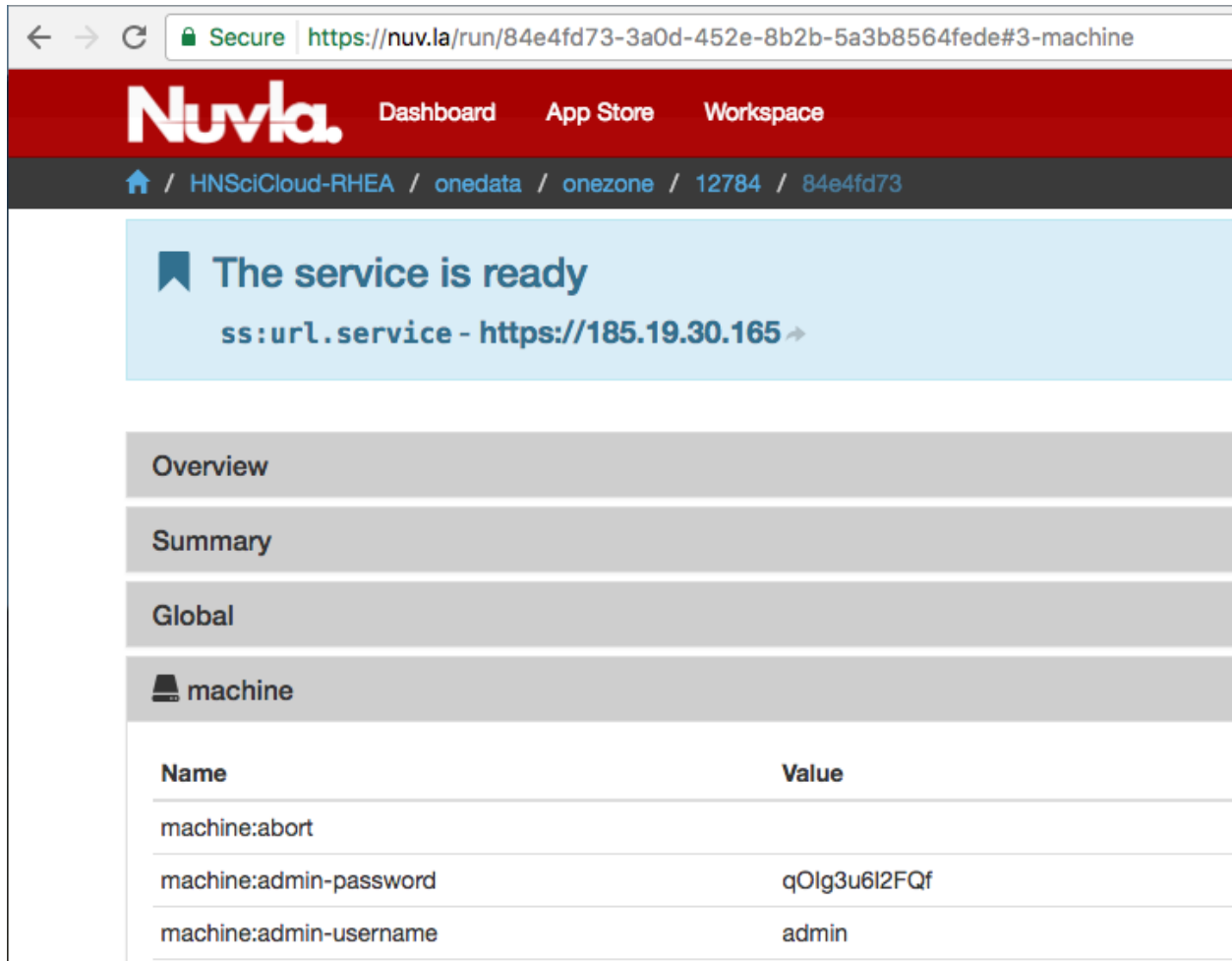
On the front page of the Onezone you should see two login options. For login in to the deployed instance of Onezone you should select the second option identified with the arrow on the image below - it uses FedIdP based authentication.



In the case above, the configuration of the integration between Onezone and project's FedIdP (<https://fed-id.nuv.la>)

was done automatically. More information on setting up of various IdPs with Onezone can be found in the Onedata's [official documentation](#).

In case extra service configuration is required, the administrator credentials can be found on the Nuvla deployment that provisioned the service under *admin-username* and autogenerated *admin-password* parameters.



The screenshot shows the Nuvla dashboard interface. At the top, there's a navigation bar with 'Nuvla.' logo and links for 'Dashboard', 'App Store', and 'Workspace'. Below this is a breadcrumb trail: 'HNSciCloud-RHEA / onedata / onezone / 12784 / 84e4fd73'. The main content area features a light blue banner stating 'The service is ready' with a link 'ss:url.service - https://185.19.30.165'. Below the banner is a sidebar with menu items: 'Overview', 'Summary', 'Global', and 'machine' (which is selected). The 'machine' section displays a table with machine parameters.

Name	Value
machine:abort	
machine:admin-password	qOlg3u6l2FQf
machine:admin-username	admin

Note: In case you need to re-deploy the Onezone instance you will have to remove *OneZoneClient* in <https://fed-id.nuv.la> that was automatically created for the previous instance of Onezone. For that

1. Login to your realm on <https://fed-id.nuv.la>.
2. Go to Clients
3. Click on *Delete* button on the row with *OneZoneClient* client

After that you should be ready to re-deploy Onezone. Don't forget to generate a new access token in Realm Settings -> Client Registration -> Initial Access Tokens.

5.1.2 GlusterFS cluster deployment from Nuvla

According to the platform architecture, Buyers are expected to be running [GlusterFS](#) based cluster on the Clouds. It is advised to deploy the respective clusters in advance before deploying Oneproviders on the Clouds.

The GlusterFS application deployment can be found Nuvla under <https://nuv.la/module/HNSciCloud/Glusterfs/gluster-cluster>

<https://nuv.la/module/HNSciCloud-RHEA/Glusterfs/gluster-cluster>

Global parameters

Scalable deployment	<input checked="" type="checkbox"/>	?
Tolerate deployment failures	<input type="checkbox"/>	?
SSH access	<input checked="" type="checkbox"/>	?
Hybrid deployment	<input checked="" type="checkbox"/>	?
Cloud	Specify for each node - € 0.0506/h	?
Cost for orchestrator(s)	exoscale-ch-gva: € 0.0076/h	?
Keep running after deployment	Always	?
Tags		?

Component parameters

node

Input volume-name	data
Input brick-path	/bricks/brick1
Initial multiplicity	2 ✓
Cloud	exoscale-ch-gva* - € 0.0430/h (2/2048/10 Small linu
Number of CPUs	2
RAM in GB	
Disk in GB	

Choose the cloud where each application component will be deployed, how many of each (parameter **Multiplicity**) and parameters for each node.

Cancel Deploy Application

To be able to manage the capacity of the cluster during its run-time, it needs to be deployed as scalable application. For that, data manager should select **Scalable deployment** option. Later the number of cluster nodes can be managed by adding and removing them via API / `ss-node-{add,remove}` CLI. More information on scalable applications in Nuvla can be found under <http://ssdocs.sixsq.com/en/latest/tutorials/ss/scalable-applications.html>. Data manager should only be concerted with scaling up/down of the provisioned GlusterFS cluster. All the scalability workflow hooks for actually scaling the cluster are already in place in the components definitions. At the same time, please consult [GlusterFS on-line documentation](#) for actual data migration between the bricks, rebalancing cluster etc. before you shrink or expand your cluster.

After that, select **Cloud** - `exoscale-ch-gva` or `open-telekom-de1`. Finally, click on *Deploy Application*.

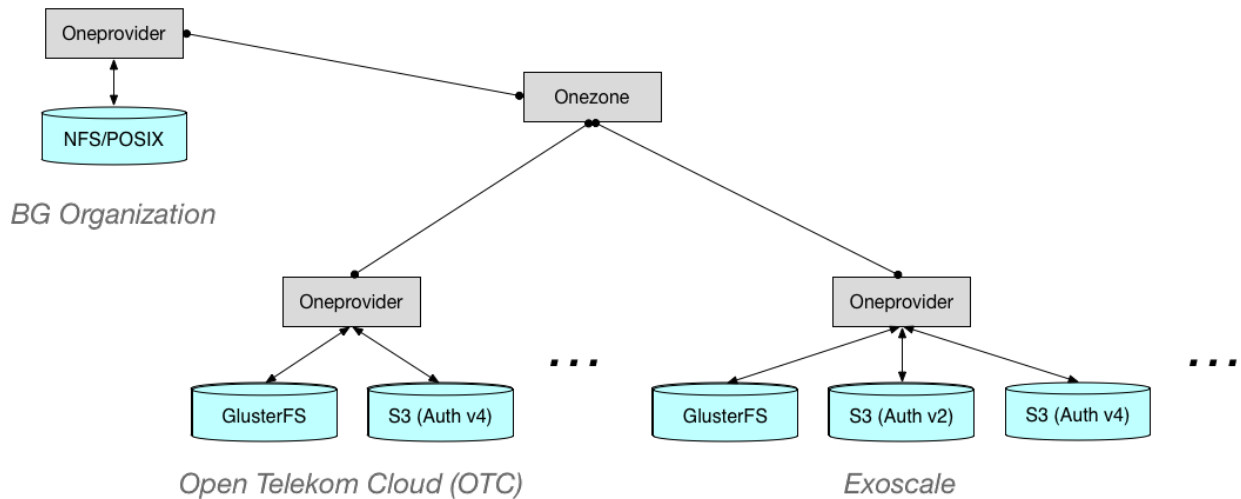
We are looking for an appropriate web GUI for monitoring and management of GlusterFS cluster that will be deployed along with the cluster to facilitate its management.

Note: After the deployment of the cluster you should copy value of the *localip* parameter of node.1 (*node.1:localip*). It will be required when configuring the backend on the Oneprovider. When scaling down the cluster, don't delete *node.1*.

Depending on the GlusterFS cluster usage strategies by the Buyer Groups the project can later create deployments of different types to cover data replication for redundancy and/or throughput. For the options available see the documentation on [GlusterFS architecture](#)

5.1.3 Oneprovider

According to the platform deployment model, the data manager must deploy at least one Oneprovider instance per Cloud (i.e., one in OTC and one in Exoscale). A single Oneprovider instance can support multiple storage backends.



Oneprovider on Cloud via Nuvla

After startup, Oneprovider service needs to register with Onezone instance. This is the reason why it should be deployed after deployment of Onezone.

On Nuvla, Oneprovider component definition can be found under <https://nuv.la/module/HNSciCloud/onedata/oneprovider>. Below is the component deployment dialog after clicking on **Deploy** in the component definition.

Deploy Application Component ✕

Global parameters Refresh

SSH access ☒ ?

Cloud ?

Tags ?

Input parameters More ▼

Input `oneprovider-version` ?

Input `onezone-host` ?

Input `onezone-ready` ?

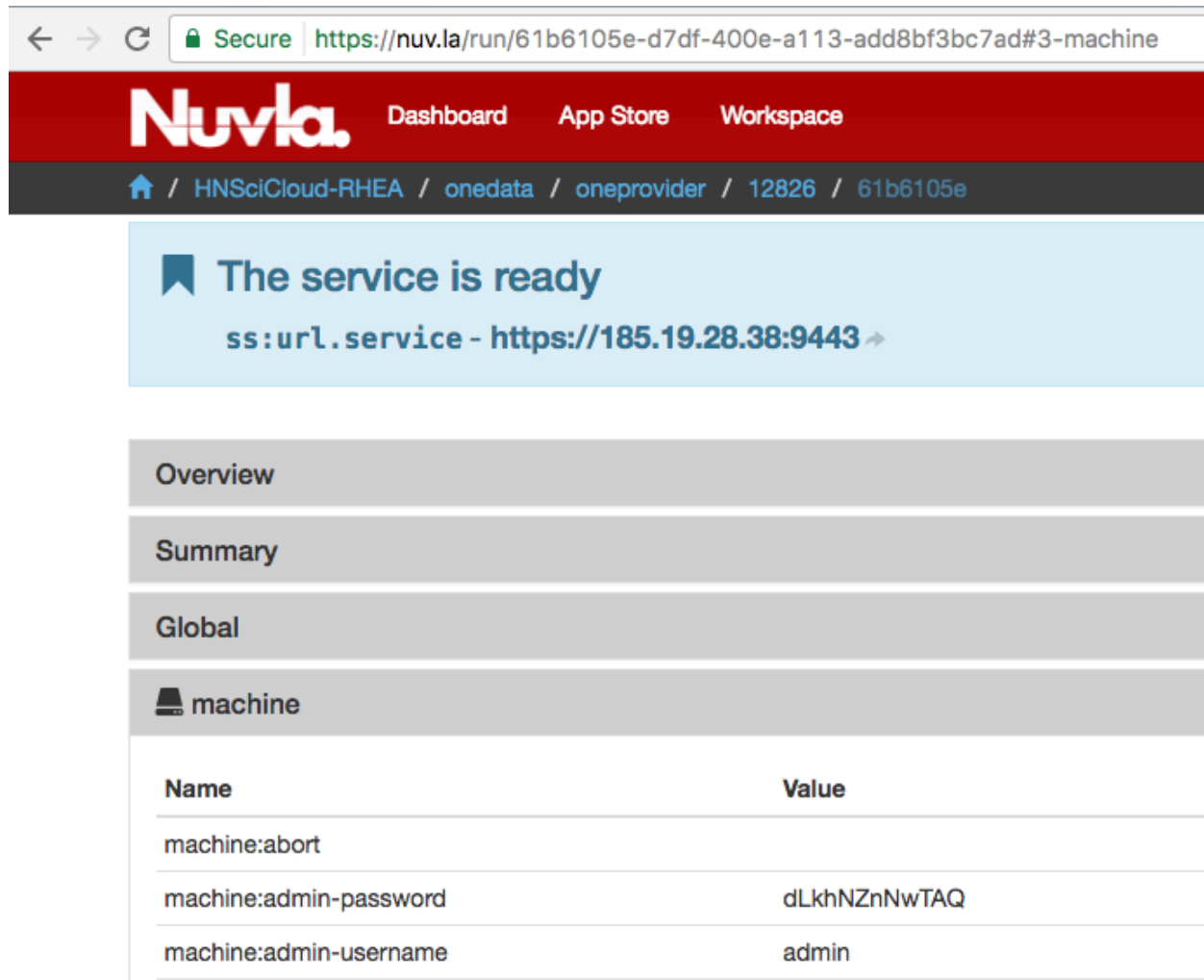
Deploy the application component `HNSciCloud-RHEA/onedata/oneprovider` (v 12826) on the selected cloud service.

First select the Cloud: `exoscale-ch-gva` or `open-telekom-de1`. Then, provide values for the input parameters.

- **onezone-host** - should be set to the IP/DNS name of the previously deployed instance of Onezone.
- **onezone-ready** - set it to `true`.
- **oneprovider-version** - leave this blank.

When parameters are set, click on *Deploy Application Component* button. After the successful deployment of Oneprovider, click on the URL defined by `ss:url.service` and you will be redirected to the running Oneprovider service.

The Oneprovider service administrator credentials can be found on the Nuvla deployment that provisioned the service under `admin-username` and autogenerated `admin-password` parameters.

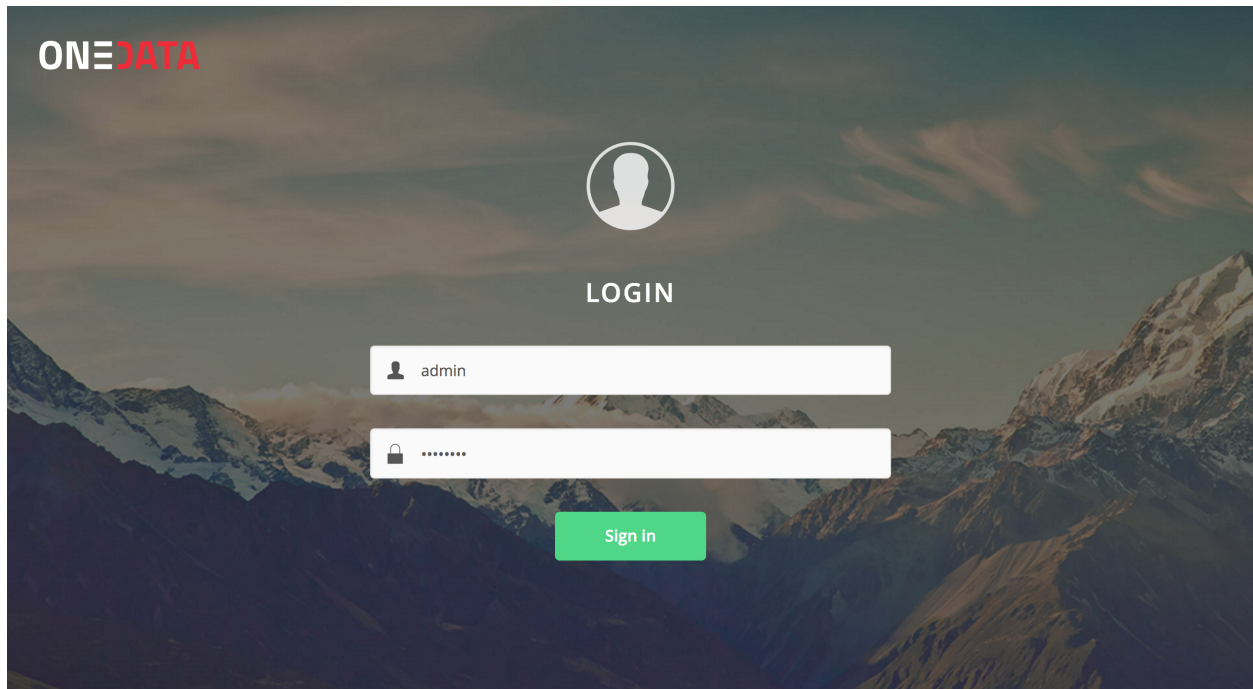


The screenshot shows the Nuvla web interface. The browser address bar displays a secure connection to `https://nuv.la/run/61b6105e-d7df-400e-a113-add8bf3bc7ad#3-machine`. The Nuvla logo is in the top left, with navigation links for Dashboard, App Store, and Workspace. A breadcrumb trail shows the path: Home / HNSciCloud-RHEA / onedata / oneprovider / 12826 / 61b6105e. A light blue banner states 'The service is ready' with a link to `ss:url.service - https://185.19.28.38:9443`. Below this, a sidebar lists navigation options: Overview, Summary, Global, and machine (selected). The main content area displays a table with machine configuration details.

Name	Value
machine:abort	
machine:admin-password	dLkhNZnNwTAQ
machine:admin-username	admin

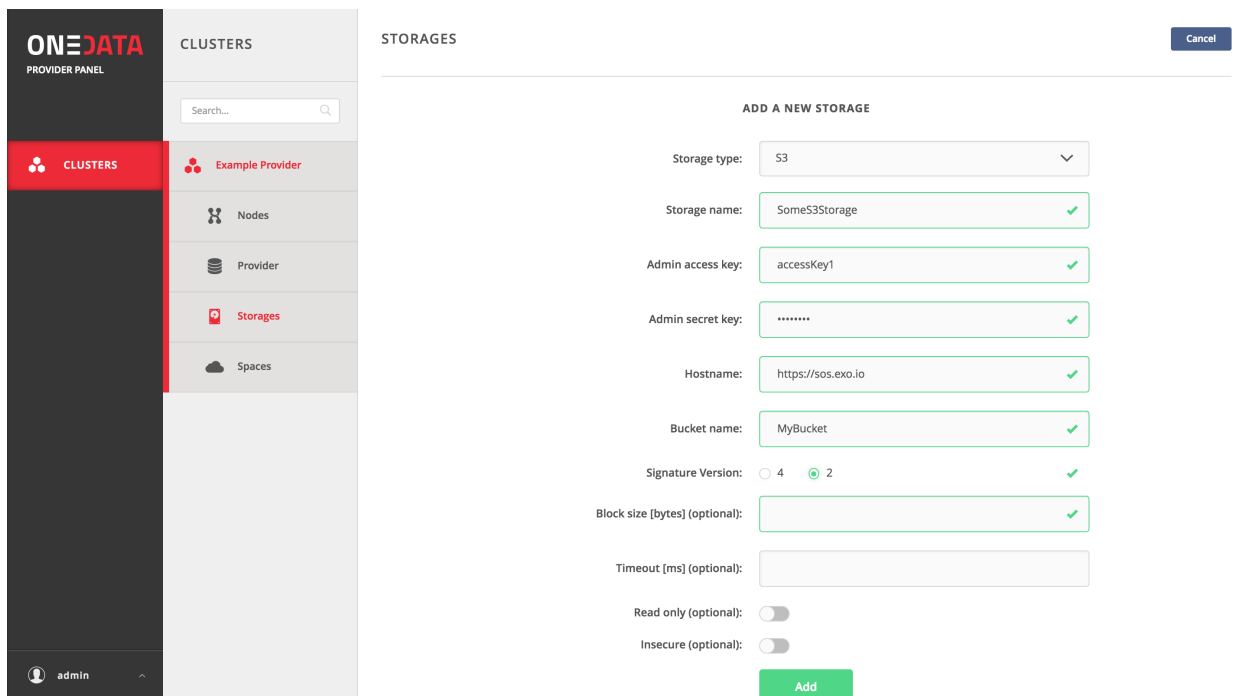
Configure Oneprovider with S3 storage type

After deploying Oneprovider VM via Nuvla, it is necessary to add an S3 storage to the Oneprovider using Onepanel administration service, running on the same host as Oneprovider. In order to open Onepanel service go to: `https://ONEPROVIDER_IP:9443` and login using administrator credentials.



After login, go to **Storages** tab and press **Add storage** button. Depending on whether the S3 bucket is on Exoscale or OTC, different configuration options must be specified:

- **Exoscale S3**
- **Hostname** - *https://sos.exo.io*
- **Signature Version** - NB! select 2 for Exoscale.



- **OTC OBS**
- **Hostname** - *https://obs.eu-de.otc.t-systems.com*

- **Signature Version** - NB! select 4 for OTC.

The screenshot shows the ONE DATA PROVIDER PANEL interface. On the left, a sidebar contains a 'CLUSTERS' menu item. The main content area is titled 'STORAGES' and features a 'Cancel' button in the top right corner. Below the title, there is a section 'ADD A NEW STORAGE' with the following fields:

- Storage type:** A dropdown menu set to 'S3'.
- Storage name:** A text input field containing 'SomeS3Storage' with a green checkmark.
- Admin access key:** A text input field containing 'accessKey1' with a green checkmark.
- Admin secret key:** A text input field containing '*****' with a green checkmark.
- Hostname:** A text input field containing 'https://obs.otc.t-systems.com' with a green checkmark.
- Bucket name:** A text input field containing 'MyBucket' with a green checkmark.
- Signature Version:** Radio buttons for '4' (selected) and '2', with a green checkmark.
- Block size [bytes] (optional):** A text input field with a green checkmark.
- Timeout [ms] (optional):** A text input field.
- Read only (optional):** A toggle switch.
- Insecure (optional):** A toggle switch.

An 'Add' button is located at the bottom right of the form.

GlusterFS Oneprovider on cloud via Nuvla

- **Volume name** - set it to the value of *volume-name* that was provided when deploying GlusterFS cluster.
- **Volume Server host** - IP address of the GlusterFS cluster. Please set it to the value from *node.1:localip* runtime parameter in the GlusterFS deployment.
- **Insecure** - this option *must be selected*. It enables the direct client access to the GlusterFS cluster.

ONEDATA PROVIDER PANEL

CLUSTERS

Search...

raw_160.44.203.212

Nodes

Provider

Storages

Spaces

admin

STORAGES Cancel

ADD A NEW STORAGE

Storage type: GlusterFS

Storage name: MyGlusterStorage ✓

Volume name: data ✓

Volume server host: 192.168.10.111 ✓

Volume server port (optional): ✓

Volume transport: ☒ tcp ☐ rdma ☐ socket

Relative mountpoint in volume (optional): ✓

Custom client translator options (optional): ✓

Timeout [ms] (optional): ✓

Insecure: ☒

Read only: ☐

Add

Oneprovider in BG organization via Nuvla

TODO

Oneprovider in BG organization manually

When deploying Oneprovider on custom storage resources it is necessary to add the storage using Onepanel administrative interface.

Currently the following storage backends are supported:

- POSIX (this includes any storage which can be mounted to Oneprovider VM such as Lustre or NFS)
- GlusterFS
- S3
- Ceph
- Openstack Swift

Each of the storage types requires different parameters to be configured properly, which can be found in the [official documentation](#).

5.1.4 One-click Deployment of Onezone and Oneproviders on Exoscale and OTC

Data manager has an option to deploy Onezone and Oneproviders on Exoscale and OTC with one button click using the following application deployment definition <https://nuv.la/module/HNSciCloud/onedata/onedata>.

Deploy Application

Global parameters

Scalable deployment

?

Tolerate deployment failures

?

SSH access

?

Cloud

Specify for each node - € 0.3930/h

?

Keep running after deployment

On Success *

?

Tags

?

Component parameters

More

oneprovider_exo

Input oneprovider-version

No value

Multiplicity

1

Cloud

exoscale-ch-gva* - € 0.0872/h (4/8192/50 Large linu

oneprovider_otc

Input oneprovider-version

No value

Multiplicity

1

Cloud

open-telekom-de1 - € 0.2186/h (4/8192/10 c2.xlarge

onezone

Input onezone-version

No value

Input hn-tenant-name

SixSq

Input hn-fedid-url

https://fed-id.nuv.la

Input hn-fedid-token

eyJhbGciOiJSUzI1NiIsImtpZCI6OiAiRi1mQkZzQjVtM

Input hn-fedid-client-id

OneZoneClient

Multiplicity

1

Cloud

exoscale-ch-gva* - € 0.0872/h (4/8192/50 Large linu

Choose the cloud where each application component will be deployed, how many of each (parameter *Multiplicity*) and parameters for each node.

Cancel

Deploy Application

When choosing **Cloud**, select *Specify for each node*. Then, select *exoscale-ch-gva* for **oneprovider_exo** and *open-*

66

Chapter 5. Data Coordinators

telekom-de1 for **oneprovider_etc**. For **onezone** component, please see *Onezone* section above for providing **hn-*** parameters. The selection of the placement of the **onezone** instance into a Cloud is up to data manager. ***-version** parameters should be left blank. Keep **Multiplicity** of the components equal to 1.

5.1.5 Managing Spaces

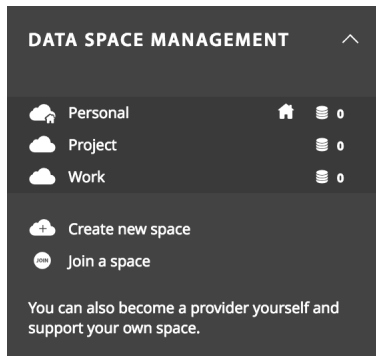
Space can be seen as a virtual directory, which contents are stored on distributed storage resources provisioned by storage providers. Each space must have at least one provider supporting it with a non-zero storage space (quota). The effective quota available to a single space is the sum of storage quotas dedicated to this space by all storage providers supporting it.

Creating spaces

Spaces in Onedata can be seen as virtual volumes or buckets, where an arbitrary directory and file hierarchy can be created.

To create a new data space follow these steps:

- In the Onezone Web Interface unfold Data space management tab located on the left menubar



- Click Create new space button
- Provide new space name in the text edit field and confirm

New space will appear in the list of spaces designated with a unique ID.

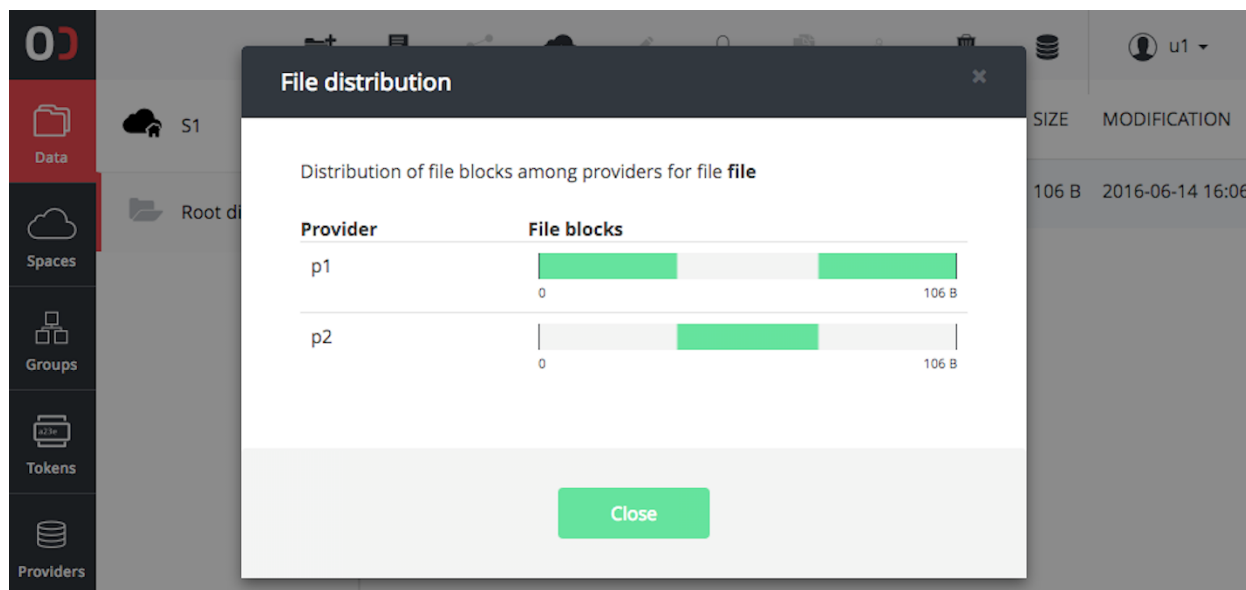
Supporting spaces with Oneprovider instances

By default new space has no storage resources associated with it. In order to add storage quota to a space, generate a space support token by clicking on *Get support* option under space name, copy the presented token and send the token to the administrator of the Oneprovider instance whose the storage resources should be assigned to this space.

5.2 Replicating Data

Onedata allows for full or partial replication of datasets between storage resources managed by Oneprovider instances. The replication can be performed at the level of entire spaces, directories, files or specific file blocks.

Onedata web interface provides visual information on the current replication of each file among the storage providers supporting the user space in which this file is located. Sample replication visualization is presented in the image below:



5.2.1 REST interface

For full control over transfer and replication users can directly invoke REST API of Oneprovider service. The documentation for this API can be found in the [official documentation](#).

5.3 Import and Export Data

5.3.1 POSIX

First of all, POSIX protocol can be used to import or export data to/from Onedata virtual filesystem using standard tools, such as 'cp' or 'rsync'. It is necessary to run the Oneclient command line tool on an access machine where the target data set is available (ingress) or where it should be exported to (egress). In case the storage managed by Onedata is available directly from the machine running Oneclient, this situation is detected automatically and the transfer between Onedata managed storage and external storage is performed directly and transparently without going via Oneprovider service, which is only called for metadata operations.

5.3.2 CDMI and REST

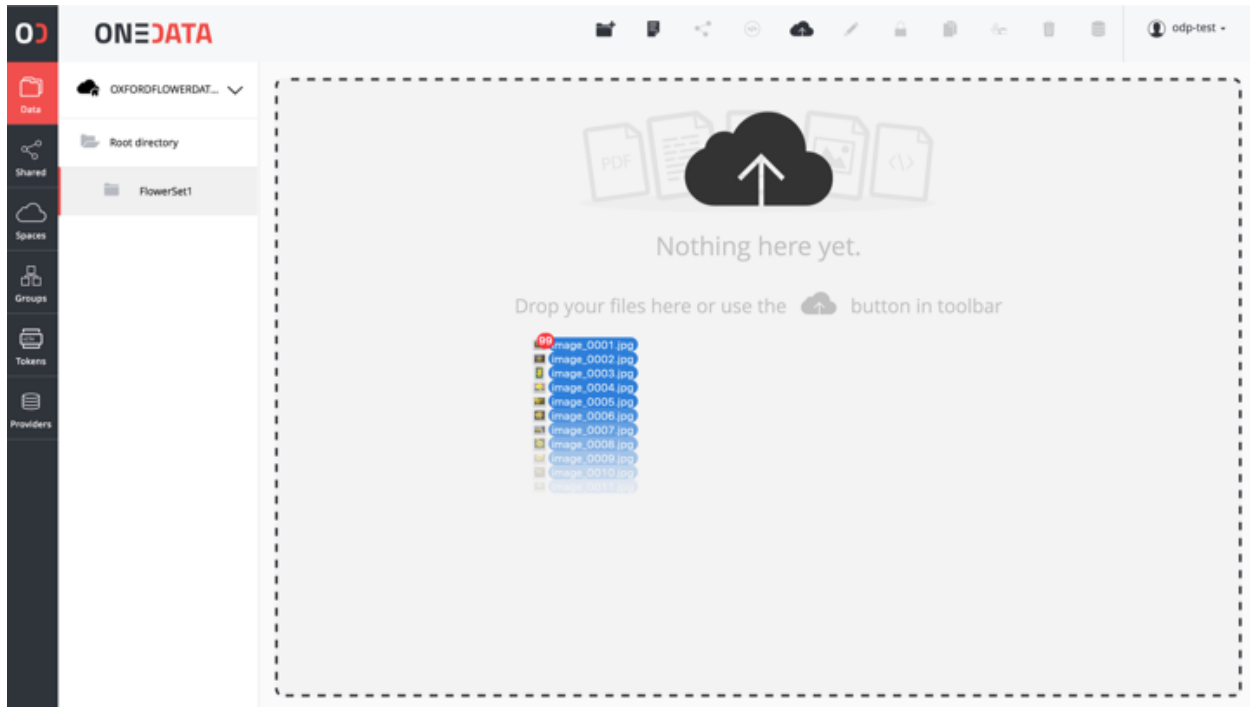
Furthermore, Onedata implements full CDMI v1.1.1 protocol specification, including data download/upload requests, and thus provides object storage system interface for all data in parallel to the POSIX protocol. This enables integration with custom user services such as portals.

For batch data transfer management, Onedata provides REST API giving programmatic access to data replication and transfer control and monitoring between the data sites.

5.3.3 GUI

Finally for small files or data sets, they can be uploaded and downloaded directly using the Web Graphical User Interface, available on all major browser and mobile devices. In order to import data using web browser, simply open

a directory within a space where the files should be uploaded, and move the files from your desktop or file browser to the Onedata Internet browser window. The upload progress will be displayed until all files are successfully uploaded.



5.3.4 Legacy data import

In use cases where there is a need to provision large legacy datasets, it is possible to configure Oneprovider service to expose such data set directly from the legacy storage without any data migration to another storage. Oneprovider service will run periodically synchronization of files on such storage, and will detect automatically new or updated files and will update its metadata database automatically. This option can be selected when adding new storage to the Oneprovider and has to be performed by Oneprovider administrators:

ONE DATA PROVIDER PANEL

CLUSTERS

Search...

ONESUPPLIER-DEMO

Nodes

Provider

Storages

Spaces

SPACES

Cancel supporting space

ADD SUPPORT FOR A SPACE

Storage: ExperimentOutput

Support token: KAJHSDLKJAHSDJKLHASJLKHjashd

Size: 100

MB GB TB

Mount in root: ☒

Import storage data: ☒

IMPORT CONFIGURATION

Import strategy: Simple scan

Max depth (optional): Example: 3

UPDATE CONFIGURATION

Update strategy: Simple scan

Max depth (optional): Example: 3

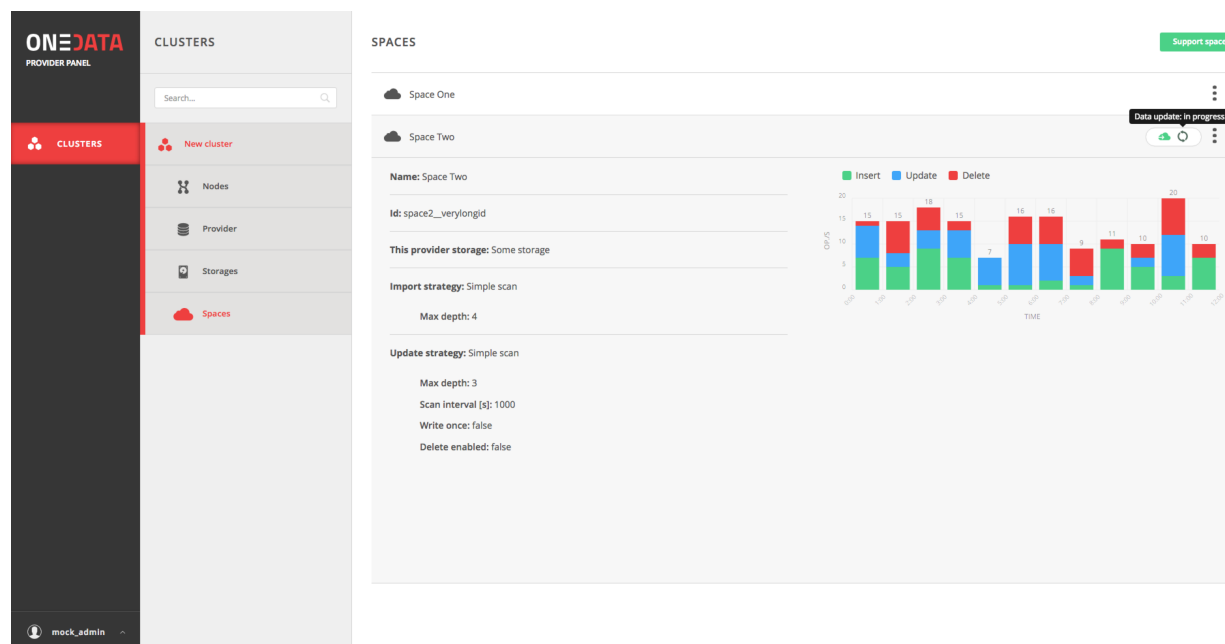
Scan interval [s]: 10

Write once: ☐

Delete enabled: ☐

Support space

Once the storage is configured for the legacy data import it will be continuously monitored for changes in the data collection (new files, modified files, deleted files) and basic statistics on the scan process will be displayed.

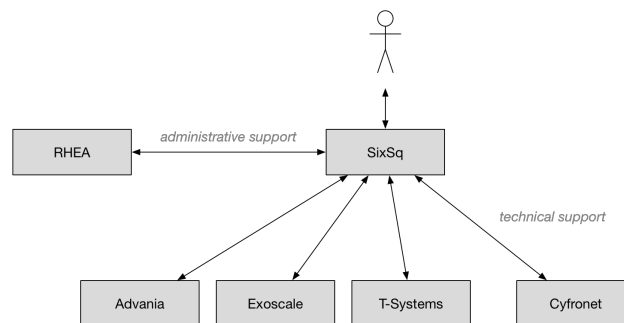


For more details, check the [official documentation](#).

Onedatify

Provisioning of legacy data using Onedata can be achieved in an easy way using a script called Onedatify, which automates the process of installing and configuring a Oneprovider instances, setting up the specified storage resources and automatically synchronizing the storage contents which are made immediately available via Onedata. Usage instructions are available [here](#).

All members of the consortium are committed to providing **timely, high-quality support** to users of the system. SixSq will act as a gateway, providing first level support itself and coordinating interactions with the other consortium members. The diagram provides an overview of the support infrastructure.



SixSq manages its support, including the support for HNSciCloud, through Freshdesk. The support services are offered in English, but may also be available in the local languages of the partners (e.g. French or German).

Users can submit support tickets via:

- Email (support@sixsq.com),
- Freshdesk, and
- Phone (+41 22 544 17 33).

A knowledge base is integrated into the Freshdesk system. A [dedicated section of the knowledge base](#) is maintained for HNSciCloud.

The consortium provides 8x5 support with a 1-hour response time. Users can set the incident priority level when submitting their tickets to ensure that they are dealt with in the appropriate time frame. Escalation of urgent or unsolved issues is possible.

7.1 Nuvla (SlipStream)

- [Nuvla Browser Interface](#)
- [General Nuvla Information](#)
- [SlipStream User Documentation](#)
- [SlipStream API Documentation](#)
- [SlipStream Python API](#)
- [SlipStream Clojure API](#)
- [SlipStream Libcloud Driver Documentation](#)

7.2 Onedata

- [Onezone Browser Interface](#) (obtain the URL from the data manager for your organization)
- [General Onedata Information](#)
- [Onedata User Documentation](#)
- [Onedata API Documentation](#)

7.3 Keycloak

- [Keycloak Portal](#)
- [General Keycloak Information](#)
- [Keycloak Documentation](#)

7.4 Advania

- [Advania Portal](#)
- [Company Website](#)

7.5 Exoscale

- [Exoscale Portal](#)
- [General Exoscale Information](#)
- [Exoscale User Documentation](#)
- [Exoscale API Documentation](#)

7.6 Open Telekom Cloud (OTC)

- [OTC Portal](#)
- [General OTC Information](#)
- [OTC User Documentation](#)
- [OTC API Reference](#)

This section contains information related to the review demonstration at DESY on 9 October 2017.

8.1 Demo

This section contains the information presented during the review demonstration that took place at DESY on 9 October 2017.

8.1.1 Prepared Services and Machines

AAI Keycloak

Group and role assignments are handled by the organization's administrator(s) via [Keycloak](#).

For this demo, we are using the **SixSq organization** (tenant). The following **groups** have been created:

- sixsq (tenant)
- sixsq/project-1 (sub-tenant)
- sixsq/project-2 (sub-tenant)

and in addition there is one **role** defined:

- data-manager

This indicates the people who will have **administrator access to Onedata services**.

The following users will be used at different points in the demo, mainly to show access rights.

tag	user	federation	IdP	groups
dm	485879@vho-switchaai...	eduGAIN	VHO	all
user-a	loomis@unitedid.org...	eduGAIN	UnitedID	sixsq, project-1
user-b	bf2ea9c63f9276...	Elixir AAI	Google	sixsq, project-2

Generation of API Key/Secret

For the browser-based interfaces to Nuvla and Onedata services, you can directly use the credentials for your Identity Provider in the eduGAIN and Elixir AAI federations.

For API and command line interface access to Nuvla, the use of **revocable API key/secret pairs are recommended**. The process for generating these key pairs can be found in the [online documentation](#) and there is a [YouTube video](#) showing the process.

[Logging into Nuvla using the REST API](#) is covered in the API documentation. For the demo, we'll use a simple script `login.sh`:

```
#!/bin/bash

url=https://nuv.la/api/session
args='--cookie-jar ~/cookies -b ~/cookies -sS'

rm -f ~/cookies

curl -XPOST $args $url \
-d href=session-template/api-key \
-d key=${KEY} \
-d secret=${SECRET} > /dev/null 2>&1

curl -XGET $args $url
```

which assumes that the key and secret are in the environment. The document returns a JSON document with your session information. This can be filtered with `jq`. For example, the command:

```
./login.sh | jq .sessions[0].roles
```

will return your roles. For the data manager role, this returns something like this:

```
"USER SixSq:uma_authorization SixSq:offline_access SixSq:/sixsq/project-1 SixSq:/
↪sixsq SixSq:data-manager SixSq:/sixsq/project-2 ANON session/b48ea165-7765-4aaf-
↪b72a-e97169ebd292"
```

notice the groups and roles defined above.

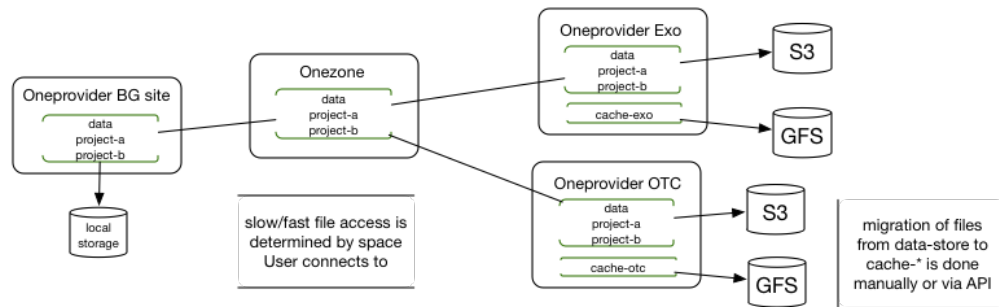
I've defined aliases for the following:

- *as-dm*
- *as-user-a*
- *as-user-b*

to make switching between users convenient at the command line.

Onedata Services

The data management infrastructure has been deployed ahead of time to save time. The deployment for the demo is shown in the following figure.



The general instructions for [configuring the data management services](#) can be found in the online documentation. The specific [Onedata deployment process](#) for this demo is shown in a YouTube video and the details are in the “data management script” document.

The deployed service endpoints and locations are given in the following table.

service	location	endpoint
Onezone	Exoscale GVA	https://159.100.243.60
Oneprovider	Exoscale GVA	https://159.100.242.90
Oneprovider	OTC	https://80.158.20.81
Oneprovider	BG (Exoscale DK)	https://159.100.247.56
GlusterFS	BG (Exoscale DK)	ssh://159.100.249.39 (alias gfs)

8.1.2 Federated AAI and Credentials (2)

- Deploy VMs with Oneclient and show that protections by groups are enforced.
- Verify that files from the BG site can be accessed via VMs in Exoscale and OTC.
- Verify that VMs cannot be accessed by non-authorized users.

Data Access Controls

Deploy a basic machine with Oneclient installed on Exoscale and OTC. To do this, use Nuvla to deploy the [Oneclient components](#).

To save time six machines have already been deployed. One on both clouds for each test user. An alias has also been provided for the GlusterFS file system on the “simulated procurer site”, which will be needed later.

user	cloud	alias	host
dm	OTC	dm-otc	80.158.21.24
dm	Exoscale	dm-exo	159.100.241.68
user-a	OTC	user-a-otc	80.158.18.127
user-a	Exoscale	user-a-exo	159.100.242.30
user-b	OTC	user-b-otc	80.158.18.171
user-b	Exoscale	user-b-exo	159.100.242.6

The Onedata spaces are mounted in the directory `/mnt/onedata`. When logging into these machines, the permissions will allow:

- The data manager to see all spaces (sixsq, project-a, project-b, cache-exo, cache-otc).
- user-a to see all spaces `_except_` project-b.

- user-b to see all spaces `_except_` project-a.

All of the data is initially on the BG site. Accessing it from any other node works transparently. In addition, this causes the file to be copied to a local cache.

Creating a file on one site makes it visible from all the other sites.

Virtual Machine Access Controls

Direct access to virtual machines is controlled through SSH. Only the keys provided by the user are added automatically to the deployed machines by Nuvla (and the underlying cloud infrastructures). Trying to access a virtual machine with the wrong SSH key will fail.

```
# wrong key
$ ssh -F /dev/null -i ~/.ssh/id_rsa_scissor 159.100.241.68

ECDSA key fingerprint is SHA256:DipROW+tGWTX/6ruqLElLH5iPLLbwGOM82fmZcWccLw.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '159.100.241.68' (ECDSA) to the list of known hosts.
Permission denied (publickey,gssapi-keyex,gssapi-with-mic).
```

When managing virtual machines through Nuvla, the service will only allow you to control the machines that you have deployed. If you share credentials, you will be able to list the machines started by others with the same credentials, but you cannot delete or change those machines.

The general access control framework used for CIMI resources will also be applied to the deployments in the future, allowing richer control over the management of deployments.

8.1.3 Transparent Data Access (1)

- Verify that User A can put data in the “simulated procurer site” directly and then access it via Onedata from a VM.
- Verify that User B can put data into a space that User A cannot access and that User B can access the file transparently, but User A cannot.

The deployment has a “simulated procurer site” deployed in the DK region of Exoscale. Machines in this site are accessible only via the WAN and not via Geant.

The “local” storage of the site is residing in a GlusterFS cluster. The machine is located at **159.100.249.39** (alias gfs). The GlusterFS data:

- Root of the “bricks” is `/bricks`.
- The local area `/bricks/brick1/local` is not shared via Onedata.
- The cloud area `/bricks/brick1/cloud` is shared.

The shared directories are mapped to spaces as follows:

- `shared` → `sixsq`
- `shared-p1` → `project-a`
- `shared-p2` → `project-b`

Now created files in all three areas via the root account on the GlusterFS node. These files should be visible (or not) in other Onedata services in accordance with the usual access controls.

```
$ ## create 1 MB files for permissions demo

$ dd bs=1024 count=1000 </dev/urandom > /bricks/brick1/cloud/shared/user-all.dat
$ dd bs=1024 count=1000 </dev/urandom > /bricks/brick1/cloud/shared-p1/user-a.dat
$ dd bs=1024 count=1000 </dev/urandom > /bricks/brick1/cloud/shared-p2/user-b.dat
```

You should see that the files are visible on the deployed Oneclient VMs and that the checksums for the individual files are the same as those on the GlusterFS node.

This showed the protection at the level of spaces. **Onedata is also capable of adding ACLs to files and directories.** This allows an even finer-grained access control.

8.1.4 Orchestration (3)

- Deploy Kubernetes via API/GUI
- Show that containers can take advantage of transparent data access
- Demonstrate a significant number of containers running within the deployed infrastructure

To emphasize, SlipStream (and Nuvla) emphasize orchestration on cloud infrastructures by encouraging the portable description of cloud applications and facilitating the automated deployment of those applications across clouds.

In addition, Nuvla contains a number of predefined applications that allow you to deploy **scalable** container-based infrastructures. Kubernetes is demonstrated here, but Docker, Docker Swarm, and Mesos are also available. Fission, a Function-as-a-Service platform based on containers is also available.

Given the size of the desired Kubernetes infrastructures, three large clusters have been deployed before the meeting. A [video of the Kubernetes deployment](#) can be found on YouTube. **The video also shows how the Kubernetes deployment can be scaled up and down.**

The pod that will be run within Kubernetes for this demonstration has:

- Oneclient installed so that the data management infrastructure can be used.
- IOPing installed to make micro-benchmarks of the data access speeds.
- Creates a benchmark resource within Nuvla for each IOPing run.
- Shows how the benchmarks can be selected for aggregating values over the benchmark documents.

The details for each step are in the following sections.

Log into the Cluster(s)

The pod will be deployed directly from the head node of each Kubernetes cluster. The IP addresses for those nodes are as follows:

cloud	vms	cores	endpoint
exoscale otc	105 105	420 420	159.100.243.172 80.158.16.116

To show what nodes are available in the cluster, do the following:

```
$ kubectl get nodes
```

There should be one line for each node and they should all be in the ‘Ready’ state.

Download the Pod Definition

The yaml file for the [pod definition](#) can be found on GitHub. Download this file.

Note that the pod definition uses the registry deployed by the consortium **registry.nuv.la**.

Set Parameters

Modify the parameters in the yaml file for the deployment:

- `OD_ACCESS_TOKEN`: Onedata access token
- `OP_HOST`: Local Oneprovider host
- `$PATH$`: Path, using `/mnt/data`
- `$CLOUD$`: cloud name
- `KEY`: Nuvla API key
- `$SECRET$`: Nuvla API secret

You'll need to provide correct Nuvla credentials to allow the upload of the benchmark resources.

Launch the Pod

The number of replicas is also a parameter in the file. The default is 1, so we'll change this to 10 initially and then something larger when everything is shown to work.

Start the pod with the following command:

```
$ kubectl create -f oneclient-io_pod.yaml
replicationcontroller "oneclient-ioping-benchmark-rc" created

$ kubectl get rc

kubectl get rc
NAME                                DESIRED    CURRENT    READY    AGE
oneclient-ioping-benchmark-rc      10         10         10       35s
```

Collect and Analyze Benchmarks

Check that the benchmarks are being created in Nuvla. A script like the following will work:

```
#!/bin/bash

curl --cookie-jar ~/cookies -b ~/cookies -sS \
  -X PUT -H 'content-type:application/x-www-form-urlencoded' \
  https://nuv.la/api/service-benchmark \
  -d '$last=5' \
  -d '$orderby=created:desc' \
  -d '$aggregation=count:id'
```

This will show the last 5 benchmark resources. You can look to see what metrics are being collected.

Then you can collect statistics over them with the following:

```
#!/bin/bash

curl --cookie-jar ~/cookies -b ~/cookies -sS \
  -X PUT -H 'content-type:application/x-www-form-urlencoded' \
  https://nuv.la/api/service-benchmark \
  -d '$last=0' \
  -d '$orderby=created:desc' \
  -d '$aggregation=count:id' \
  -d '$aggregation=avg:ioping:iops' \
  -d '$aggregation=avg:ioping:bytes_per_second'
```

You can filter this to get reasonable output with `jq`:

```
$ ./benchmarks-stats.sh | jq .aggregations
{
  "count:id": {
    "value": 1189
  },
  "avg:ioping:bytes_per_second": {
    "value": 2868820.260188088
  },
  "avg:ioping:iops": {
    "value": 2.7460815047021945
  }
}
```

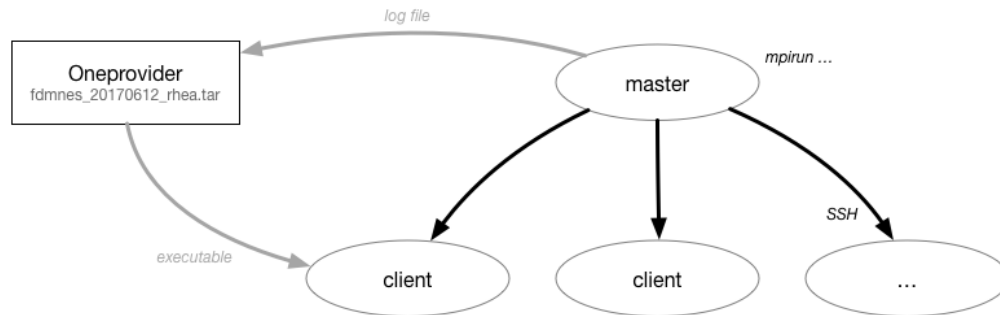
With the information in the benchmark much more detailed information could be obtained concerning the latencies and bandwidths between various points in the system.

8.1.5 HPCaaS (7)

- Use of FDMNES, an OpenMPI-based HPC application.
- Deployment of cluster of different sizes via SlipStream Libcloud driver
- Transparently access input files and executables via Onedata
- Publishing of result log to Onedata infrastructure
- Show resources used during deployment with SlipStream monitoring infrastructure
- Show resource utilization for individual deployments with SlipStream metering (accounting) resources

FDMNES

The example HPC application is the FDMNES application from ESRF, which uses OpenMPI over a cluster of compute nodes. The following figure shows the deployed components and the primary interactions between them.



By default, there will be 1 master and 2 client nodes deployed. On Exoscale the default size is “Large” (4 CPU, 8 GB RAM) and on OTC the default size is “c1.large” (4 CPU, 4 GB RAM).

As for most example applications for HNSciCloud, a [SlipStream application](#) has been created for this deployment.

Deploy with Libcloud

This will be deployed and controlled via the [SlipStream Libcloud driver](#).

The following script will be used to deploy the FDMNES application. The default configuration will be deployed on both clouds and then larger one with more and larger machines.

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

#
# convenience modules
#
import os
from pprint import pprint as pp

#
# require modules for the slipstream driver
#
import slipstream.libcloud.compute_driver
from libcloud.compute.providers import get_driver

#
# create the driver itself
#
slipstream_driver = get_driver('slipstream')

#
# log into Nuvla using an API key/secret
# API key and secret are taken from the environment
#
ss = slipstream_driver(os.environ["KEY"],
                      os.environ["SECRET"],
                      ex_login_method='api-key')

#
# deploy a k8s cluster with defined number of nodes
# on the specified cloud
#
image = ss.get_image('HNSciCloud/fdmnes/fdmnes-demo')

#
```

```

# change the cloud (location) here
#
connector = os.environ["CONNECTOR"]
location = filter((lambda x: x.id==connector), ss.list_locations())[0]
pp(location)

#
# change the size here
#

offer = None

# 12/64/50 Mega Exoscale
#offer="service-offer/25e47696-bd4f-481d-9352-dcebe087a3de"

# 16/64/200 OTC
#offer="service-offer/7b7fb2a4-ec2d-43b0-81ec-e284f05257c7"

pp(ss.list_sizes(location))

if offer is not None:
    node_size = filter((lambda x: x.id==offer), ss.list_sizes(location))[0]
else:
    node_size = None

pp(node_size)

access_token = os.environ["ONEDATA_TOKEN"]
provider_hostname = os.environ["ONEDATA_OP_HOST"]
onedata_space = os.environ["ONEDATA_SPACE"]

multiplicity = None
#multiplicity = {"master": 1, "client": 4}

params = {"master": {"access-token": access_token,
                    "provider-hostname": provider_hostname,
                    "onedata-space": onedata_space},
          "client": {"access-token": access_token,
                    "provider-hostname": provider_hostname,
                    "onedata-space": onedata_space}}

node = ss.create_node(image=image,
                      size=node_size,
                      location=location,
                      ex_multiplicity=multiplicity,
                      ex_parameters=params,
                      ex_keep_running='never',
                      # ex_tolerate_failures=allowed_failures,
                      ex_scalable=False)

print node.id

#
# terminate the cluster when finished
#
#ss.destroy_node(node)

```

Transparent Data Access

The input data files and executable are taken from Onedata. You can see the input tarball (with the input files and executable) in the directory `data/fdmnes`.

The results logs will appear in `data/fdmnes/logs`. They have a naming that shows the cloud (IP Address), start time, slots, and execution time.

Current Usage for Deployment

We can use the SlipStream REST API to show the current resource utilization, in general and by deployment. The following script shows how to use the CIMI filtering capabilities to select resources and then to aggregate the values.

```
#!/bin/bash

deployment_id=$1

curl --cookie-jar ~/cookies -b ~/cookies -sS \
  -X PUT -H 'content-type:application/x-www-form-urlencoded' \
  https://nuv.la/api/virtual-machine \
  -d '$filter=deployment/href="run/'"${deployment_id}"'"' \
  -d '$last=0' \
  -d '$aggregation=count:id' \
  -d '$aggregation=sum:serviceOffer/resource:vcpu' \
  -d '$aggregation=sum:serviceOffer/resource:ram'
```

As you can see from the URL, this works for virtual machine resources. This will be extended to data and other resources in the future.

The result returns a JSON document. The interesting content is in the “aggregations” key.

```
$ ./usage.sh c7cccb79-7156-4f39-9e3d-6db5272034e1 | jq .aggregations
{
  "sum:serviceOffer/resource:vcpu": {
    "value": 13
  },
  "count:id": {
    "value": 4
  },
  "sum:serviceOffer/resource:ram": {
    "value": 25088
  }
}
```

Metering (Accounting) Information

Historical information about resource usage is kept by the “metering” resources. The API for the metering works nearly identically to that for the resource utilization. Simply the URL changes.

Use the following script to see the resource utilization for a given deployment.

```
#!/bin/bash

deployment_id=$1

curl --cookie-jar ~/cookies -b ~/cookies -sS \
```



```
-X PUT -H 'content-type:application/x-www-form-urlencoded' \
https://nuv.la/api/metering \
-d '$filter=deployment/href="'${deployment_id}'' \
-d '$last=0' \
-d '$aggregation=count:id' \
-d '$aggregation=sum:price' \
-d '$aggregation=sum:serviceOffer/resource:vcpu' \
-d '$aggregation=sum:serviceOffer/resource:ram'
```

And the results of this are visible in the returned JSON file.

```
$ ./metering.sh c7cccb79-7156-4f39-9e3d-6db5272034e1 | jq .aggregations
{
  "sum:serviceOffer/resource:vcpu": {
    "value": 106
  },
  "count:id": {
    "value": 34
  },
  "sum:serviceOffer/resource:ram": {
    "value": 201728
  },
  "sum:price": {
    "value": 0.027280366433333335
  }
}
```

8.1.6 Reporting and Accounting (6)

- Show current usage by tenant, subtenant, and user.
- Show historical usage by tenant, subtenant, and user.
- Show how quotas are defined, evaluated, and enforced.

Overview

The mechanisms for monitoring the current resource utilization, accounting for usage over time, and enforcing limits on resource usage are closely related.

The infrastructure to support these functions works like this:

- Probes maintain a representation of the current global state of the system in a set of documents.
- The metering infrastructure takes frequent, regular snapshots of the global state, augmenting the information with, for example, pricing information.
- Quotas compare the current global state with a resource request to verify that a request is within the stated limits.
- “Billing” selects subsets of the metering documents to produce time-based reports by tenant, sub-tenant, user, etc.

The mechanisms are completely general and can be applied to any resource for which there is a probe. Currently only a probe for virtual machines is implemented, but probes for storage are planned.

Current Usage

As you've seen in the HPCaaS demonstration, the current resource usage can be obtained by selecting resource documents and aggregating over values.

The filtering can take into account any field in the resource document, including tenant, user, etc.

For example to see my current usage, I can use a query like the following:

```
#!/bin/bash

user_id="$1"
cloud_id="$2"

curl --cookie-jar ~/cookies -b ~/cookies -sS \
  -X PUT -H 'content-type:application/x-www-form-urlencoded' \
  https://nuv.la/api/virtual-machine \
  --data-urlencode '$filter=deployment/user/href="user/'"$user_id"'"' \
  -d '$last=0' \
  -d '$aggregation=count:id' \
  -d '$aggregation=sum:serviceOffer/resource:vcpu' \
  -d '$aggregation=sum:serviceOffer/resource:ram'

# add for cloud filtering
# --data-urlencode '$filter=connector/href="connector/'"$cloud_id"'"' \
```

```
$ ./current-usage.sh '485879@vho...' | jq .aggregations
{
  "sum:serviceOffer/resource:vcpu": {
    "value": 905
  },
  "count:id": {
    "value": 227
  },
  "sum:serviceOffer/resource:ram": {
    "value": 1437696
  }
}
```

To show the value for all cloud and all visible resources, remove the filter on the `deployment/user/href` value. To see for only a particular cloud add the filter on the `connector/href` value.

Historical Usage

Historical views of the resource utilization can be obtained in an analogous way from the metering resources. The script looks very similar, but there are a few of differences:

- The URL changes to “<https://nuv.la/api/metering>”.
- The `snapshot-time` field is added to allow for selection of arbitrary time periods.
- The `price` field is added when the resource price information is known, allowing a calculation of the approximate cost of resources.

So concretely, the filter can select on time and the aggregations can now include price.

```
#!/bin/bash

user_id="$1"
```

```
cloud_id="$2"

curl --cookie-jar ~/cookies -b ~/cookies -sS \
  -X PUT -H 'content-type:application/x-www-form-urlencoded' \
  https://nuv.la/api/metering \
  --data-urlencode '$filter=deployment/user/href="user/'${user_id}'"' \
  -d '$filter=snapshot-time>="2017-10-01T00:00:00.000Z"' \
  -d '$filter=snapshot-time<="2017-10-08T00:00:00.000Z"' \
  -d '$last=0' \
  -d '$aggregation=count:id' \
  -d '$aggregation=sum:price' \
  -d '$aggregation=sum:serviceOffer/resource:vcpu' \
  -d '$aggregation=sum:serviceOffer/resource:ram'

# add for cloud filtering
# --data-urlencode '$filter=connector/href="connector/'${cloud_id}'"' \
```

```
$ ./historical-usage.sh '485879@vho...' | jq .aggregations
{
  "sum:serviceOffer/resource:vcpu": {
    "value": 34061
  },
  "count:id": {
    "value": 30228
  },
  "sum:serviceOffer/resource:ram": {
    "value": 94775808
  },
  "sum:price": {
    "value": 94.92273246906117
  }
}
```

Quotas

Quotas place limits on resource utilization. For the hybrid cloud system, each quota is a separate document that describes

- The type of resource to use (i.e. the resource collection),
- A filter to select the documents to include,
- The field/method on which to aggregate, and
- A numerical limit.

The quota resource also contains an action called `collect` that will calculate the current values for the quota. The URL for this action is included in the quota document itself. The returned document is the quota document augmented with “currentAll” and “currentUser” fields.

The “currentAll” field includes all resources falling under the given quota, whereas the “currentUser” provides the user’s contribution to the “currentAll” value. The limit is always compared to the “currentAll” value.

Any quota resource that is visible to a user is applied to that user. **That is, the resource ACL determines who the quota to whom applies.**

You can list the visible quotas with a simple command:

```
$ curl --cookie-jar ~/cookies -b ~/cookies -sS \
-X GET \
https://nuv.la/api/quota
```

using your credentials. This will list all quotas that apply to you.

Concretely a quota for limiting the number of virtual machines deployed by an individual user would look like (with uninteresting fields stripped):

```
{
  "name" : "RAM (VHO)",
  "description" : "limits total RAM usage for VHO account",

  "resource" : "VirtualMachine",
  "selection" : "deployment/user/href='user/485879@vho-switchaai.chhttps://aai-
↳logon.vho-switchaai.ch/idp/shibboleth!https://fed-id.nuv.la/samlbridge/module.php/
↳saml/sp/metadata.php/sixsq-saml-bridge!uays4u2/dk2qefyxzsv9uiicv+y=' "

  "aggregation" : "sum:serviceOffer/resource:ram",
  "limit" : 10240,

  "acl" : {
    "owner" : { "...": "..." },
    "rules" : [
      { "...": "..." },
      {
        "principal" : "485879@vho-switchaai.chhttps://aai-logon.vho-switchaai.ch/idp/
↳shibboleth!https://fed-id.nuv.la/samlbridge/module.php/saml/sp/metadata.php/sixsq-
↳saml-bridge!uays4u2/dk2qefyxzsv9uiicv+y=",
        "right" : "VIEW",
        "type" : "USER"
      }
    ]
  },
  "operations" : [ {
    "rel" : "http://sixsq.com/slipstream/1/action/collect",
    "href" : "quota/6448b707-eae3-4646-878c-6dce3682a526/collect"
  } ],
}
```

By sending an HTTP POST request to the (relative) “collect” URL, you can get the current usage information related to this quota.

```
$ curl --cookie-jar ~/cookies -b ~/cookies -sS \
-X POST \
https://nuv.la/api/quota/6448b707-eae3-4646-878c-6dce3682a526/collect | \
jq .currentAll,.currentUser,.limit
1437696
1437696
10240
```

This shows that I’ve completely blown my quota!

The next release of SlipStream will include the ability to parameterize the quotas and to filter directly over groups.

8.1.7 Thanks!

Just a reminder that the [main documentation](#) can be found in ReadTheDocs and that there is a [knowledge base](#) in Freshdesk.

Feedback on improving and expanding the documentation is welcome!

8.2 Onedata Deployment

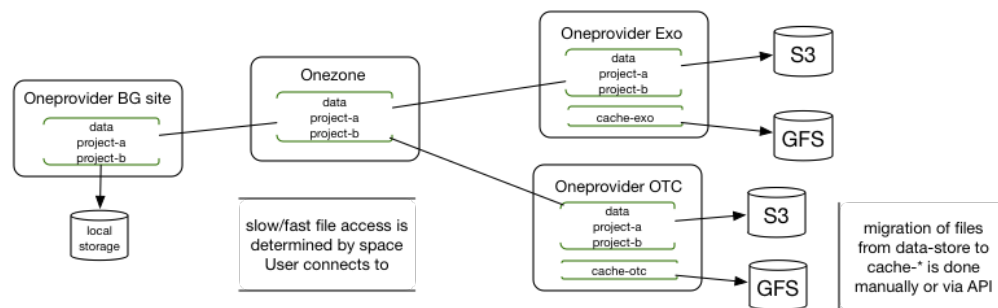
The following demonstrates the deployment of Onedata based Data Management infrastructure for HNSciCloud. The deployment is done on three cloud sites: OTC, Exoscale GVA and Exoscale DK.

A video of the [Onedata deployment process](#) can be found on YouTube.

We will deploy and configure:

- Onezone in Exoscale GVA
- Oneprovider and GlusterFS in OTC
- Oneprovider and GlusterFS in Exoscale GVA
- Oneprovider and GlusterFS in Exoscale DK

Everything deployed in Exoscale DK is viewed as Buyers Group simulated site.



[Nuvla service](#) will be used for all the deployments.

8.2.1 Prerequisites

1. Create the following buckets `sixsq-data`, `sixsq-project-1`, `sixsq-project-2` in object storages of
 - Exoscale <https://portal.exoscale.ch/storage>
 - OTC <https://auth.otc.t-systems.com/authui/login#/login> -> Object Storage Service

Later the buckets will correspondingly be mapped to `data`, `project-1` and `project-2` spaces in Onedata.

2. Configure groups, roles and Data Manager user in KeyCloak.

- **Groups**
 - `/sixsq` set as default in the realm
 - `/sixsq/project-1`
 - `/sixsq/project-2`
- **Roles:** `data-manager`

- Assign the groups and the role to the Data Manager user.

8.2.2 Deployment

1. Deploy Onedata on two cloud sites - Exoscale GVA and OTC - using <https://nuv.la/module/HNSciCloud/onedata/onedata>. This is the deployment of the standard Onedata stack for two cloud sites.

- Onedata on Exoscale GVA
- Oneprovider on Exoscale GVA
- Oneprovider on OTC

Documentation for deployment <http://hn-prototype-docs.readthedocs.io/en/latest/data-manager/service-deployment.html#one-click-deployment-of-onezone-and-oneproviders-on-exoscale-and-otc>

2. Deploy GlusterFS using <https://nuv.la/module/HNSciCloud/GlusterFS/gluster-cluster> on

- Exoscale GVA (VM size 2/4/100)
- OTC (VM size 2/4/100)
- Exoscale DK (VM size 2/4/400)

Documentation for deployment <http://hn-prototype-docs.readthedocs.io/en/latest/data-manager/service-deployment.html#glusterfs-cluster-deployment-from-nuvla>

3. After deployment in step 1. is ready, deploy Oneprovider BG <https://nuv.la/module/HNSciCloud/onedata/oneprovider> for BG site.

Documentation for deployment <http://hn-prototype-docs.readthedocs.io/en/latest/data-manager/service-deployment.html#oneprovider>

4. As Data Manager, authenticate to Onedata provisioned in step 1. Create the following set of spaces that will be used by the project.

- data
- project-1
- project-2
- cache-exo
- cache-otc

5. Populate GlusterFS BG with test data: 3 x 2000 50MB files.

- On GlusterFS BG instance run the following script

```
#!/bin/bash

# Local sub-tree not shared with cloud.
mkdir -p /bricks/brick1/local
date > /bricks/brick1/local/test-file.txt

# Sub-tree shared with cloud.
# Oneprovider will be mounting spaces into 'shared*/' sub-directories.

base=/bricks/brick1/cloud

# cloud/
#     shared/
#         d-data/test-{1..2000}.file
```

```
#      shared-p1/
#          d-p1/test-{1..2000}.file
#      shared-p2/
#          d-p2/test-{1..2000}.file

for d in shared/d-data shared-p1/d-p1 shared-p2/d-p2; do
    dir=$base/$d
    mkdir -p $dir
    echo == $dir ==
    for n in {1..2000}; do
        dd if=/dev/zero of=$dir/test-$n.file bs=50MB count=1 &>/dev/null
    done
    for n in {1..2000}; do
        date +%s%N >> $dir/test-$n.file
    done
done
```

6. Configure storages and spaces on Oneprovider BG.

- Give provider a friendly name: `OP-SixSq-site`
- **Add GlusterFS BG storage**
 - storage name: `gfs-bg-shared`
 - volume name: `data`
 - Relative mountpoint in volume: `cloud/shared/`
- **Support data space by `gfs-bg-shared` storage**
 - Get space support request token from Oneprovider
 - Support space: `Insecure on, Mount in root on, Import storage data with continuous update.`
- **Add GlusterFS BG storage**
 - storage name: `gfs-bg-p1`
 - volume name: `data`
 - Relative mountpoint in volume: `cloud/shared-p1/`
- **Support `project-1` space by `gfs-bg-p1` storage**
 - Get space support request token from Oneprovider
 - Support space: `Insecure on, Mount in root on, Import storage data with continuous update.`
- **Add GlusterFS BG storage**
 - storage name: `gfs-bg-p2`
 - volume name: `data`
 - Relative mountpoint in volume: `cloud/shared-p2/`
- **Support `project-2` space by `gfs-bg-p2` storage**
 - Get space support request token from Oneprovider
 - Support space: `Insecure on, Mount in root on, Import storage data with continuous update.`
- **Validation.**
 - Go to Onezone and refresh the main page. You should see provider `OP-SixSq-site`.

- Go to the new provider and then to files. You should see the directories with files under all the spaces: `data`, `project-1`, `project-2`. The files are on the `OP-SixSq-site` and provided by GlusterFS.

7. Configure storages and spaces on Oneprovider EXO.

- Give provider a friendly name: `OP-Exoscale`
- **Add S3 EXO storage**
 - storage name: `s3-exo-data`
 - bucket name: `sixsq-data`
 - S3 server: `sos.exo.io`
- **Support data space by s3-exo-data storage**
 - Get space support request token from Oneprovider
 - Support space: `Insecure on, Mount in root on`
- **Add S3 EXO storage**
 - storage name: `s3-exo-p1`
 - bucket name: `sixsq-project-1`
 - S3 server: `sos.exo.io`
- **Support project-1 space by s3-exo-p1 storage**
 - Get space support request token from Oneprovider
 - Support space: `Insecure on, Mount in root on`
- **Add S3 EXO storage**
 - storage name: `s3-exo-p2`
 - bucket name: `sixsq-project-2`
 - S3 server: `sos.exo.io`
- **Support project-2 space by s3-exo-p2 storage**
 - Get space support request token from Oneprovider
 - Support space: `Insecure on, Mount in root on`
- **Add GlusterFS EXO storage**
 - storage name: `gfs-exo`
 - volume name: `data`
- **Support cache-exo space by gfs-exo storage**
 - Get space support request token from Oneprovider
 - Support space: `Insecure on, Mount in root on`
- **Validation.**
 - Go to Onezone and refresh the main page. You should see provider `OP-Exoscale`.
 - You should see the provider supports the following spaces: `data`, `project-1`, `project-2`, `cache-exo`.

8. Configure storages and spaces on Oneprovider OTC.

- Give provider a friendly name: OP-OTC
 - **Add S3 OTC storage**
 - storage name: s3-otc-data
 - bucket name: sixsq-data
 - S3 server: obs.eu-de.otc.t-systems.com
 - **Support data space by s3-otc-data storage**
 - Get space support request token from Oneprovider
 - Support space: Insecure on, Mount in root on
 - **Add S3 OTC storage**
 - storage name: s3-otc-p1
 - bucket name: sixsq-project-1
 - S3 server: obs.eu-de.otc.t-systems.com
 - **Support project-1 space by s3-otc-p1 storage**
 - Get space support request token from Oneprovider
 - Support space: Insecure on, Mount in root on
 - **Add S3 OTC storage**
 - storage name: s3-otc-p2
 - bucket name: sixsq-project-2
 - S3 server: obs.eu-de.otc.t-systems.com
 - **Support project-2 space by s3-otc-p2 storage**
 - Get space support request token from Oneprovider
 - Support space: Insecure on, Mount in root on
 - **Add GlusterFS OTC storage**
 - storage name: gfs-otc
 - volume name: data
 - **Support cache-otc space by gfs-otc storage**
 - Get space support request token from Oneprovider
 - Support space: Insecure on, Mount in root on
 - **Validation.**
 - Go to Onezone and refresh the main page. You should see provider OP-OTC.
 - You should see the provider supports the following spaces: data, project-1, project-2, cache-otc.
9. Share storage spaces with user groups.
- As Data Manger user login to Onezone.
 - Add /sixsq group into data, cache-exo and cache-otc spaces.
 - Add /sixsq/project-1 group into project-1 space.

- Add `/sixsq/project-2` group into `project-2` space.

8.3 YouTube Videos

- API key/secret generation
- Onedata deployment process
- Kubernetes deployment

This section contains information related to the training on 17 April 2018 for both end-users and procurers. **This section, and references to the main body of the documentation, constitutes Deliverable D-PIL-3.2.**

9.1 End-Users Training

9.1.1 Introduction

This training will cover three core activities for users of the RHEA Consortium's hybrid cloud infrastructure:

1. Deploying cloud applications via Nuvla,
2. Taking advantage of the Libcloud API for multi-cloud management, and
3. using Onedata services for managing data.

After this training, the users should have a general idea of how these activities can be accomplished on the platform. Further details can be found in the rest of the [platform documentation](#).

Prerequisites

To follow the complete end-user tutorial, you must have the following:

- A laptop with access to the Internet (probably via wifi),
- An SSH client installed on your laptop (Linux/Mac OS: installed by default, Windows: use [Putty](#)),
- An **OpenSSH** keypair ([Linux/Mac OS](#), [Windows](#)), and
- An identity from a procurer's organization that can be used through the eduGAIN or Elixir AAI identity federations.

If you don't have these, consult the referenced documentation or ask your administrator.

Logging in to Nuvla

The tutorial will use a set of student accounts that have been created ahead of time.

- To log in to Nuvla, visit <https://nuv.la> with your browser.
- Click the *Log in* button.
- On the next popup window select the “SixSq Realm” next to the “SixSq/RHEA Federated Login” button.
- Click on the “SixSq/RHEA Federated Login” button.

At this point, you will be redirected to the Keycloak server that allows Federated Identity management via the eduGAIN and Elixir AAI identity federations.

From the Keycloak login page:

- Enter your student credentials (username=`studentXX`, password=`training`).
- Click the *Log In* button.

At the end of the process you will be redirected to the App Store page of Nuvla and offered a tutorial. You can click away the tutorial dialog, as we’ll not be using that here.

Setting SSH keys

So that you can log into virtual machines that you deploy, Nuvla must have a copy of your public SSH key. To add your public SSH public key to Nuvla:

- Go to the upper right menu with your *StudentXX* username and choose “Profile”.
- Click the *Edit* button.
- Open the *General Section* by clicking on the section header.
- In the Default cloud list, choose *exoscale-ch-gva*.
- Copy your SSH public key (in Linux usually in `~/.ssh/id_rsa.pub`) to the *SSH Public Key(s) (one per line)* text area.
- Click the *Save* action.

Be sure to copy the full contents of your public SSH key as a **single line of text**!

9.1.2 Deploying on Nuvla

Deploying a Simple Component

We will start by deploying a simple application component. A “component” for SlipStream is a single-VM deployment. The example we use here is a minimal CentOS image with the Libcloud library (and dependencies) installed.

- When logged into Nuvla, open the [Centos-libcloud](#) application
- Click on the *Deploy* action.
- Click on the *Deploy Application Component* button after optionally providing tags.

On the “Deploy Application Component” dialog you may optionally set the Tag field with comma separated values to help you remember why the component was deployed. Because we will later use it when presenting the *Managing VMs with Libcloud* feature, you may want to set the tags “libcloud, training”.

At the end of the process, you’ll be redirected to the Nuvla Dashboard. At the bottom, you should be able to see the “Centos-libcloud” component being deployed. This will take a couple of minutes to complete.

Deploying an Application

For SlipStream an “application” consists of a deployment with multiple virtual machines. To assist with the lifecycle management of applications, Nuvla will deploy one “orchestrator” machine per cloud.

To deploy, the example application:

- Ensure that you are logged into Nuvla.
- Navigate to the [Docker Swarm](#) application or choose this application from the App Store.
- Click on the *Deploy* action.
- Click on the *Deploy Application* button in the dialog.

You should not need to change anything in the deployment dialog, although, as before, you may add tags to your deployment.

You will again be redirected to the Nuvla Dashboard at the end of the process. This application will run a Docker Swarm cluster with one Master and one Worker by default; you can change the number of workers in the deployment dialog if you want. This will also take a few minutes to complete.

If you have time, you can log into the master node and run a container on this swarm.

9.1.3 Managing VMs with Libcloud

From your Dashboard, identify the Centos-libcloud component you have previously deployed (see Section [Deploying a Simple Component](#)) and **click on its ‘Service URL’ link at the top of the page**. This will log you into your virtual machine via SSH.

Warning: If you’ve not configured your browser to open SSH links automatically, you will need to open a terminal (or SSH client) and log in manually, using the information in the link and your SSH public key.

Using the Libcloud Compute Driver for SlipStream

For the browser-based interfaces to Nuvla and Onedata services, you can directly use the credentials for your Identity Provider in the eduGAIN and Elixir AAI federations.

For API and command line interface access to Nuvla, the use of revocable API key/secret pairs are required.

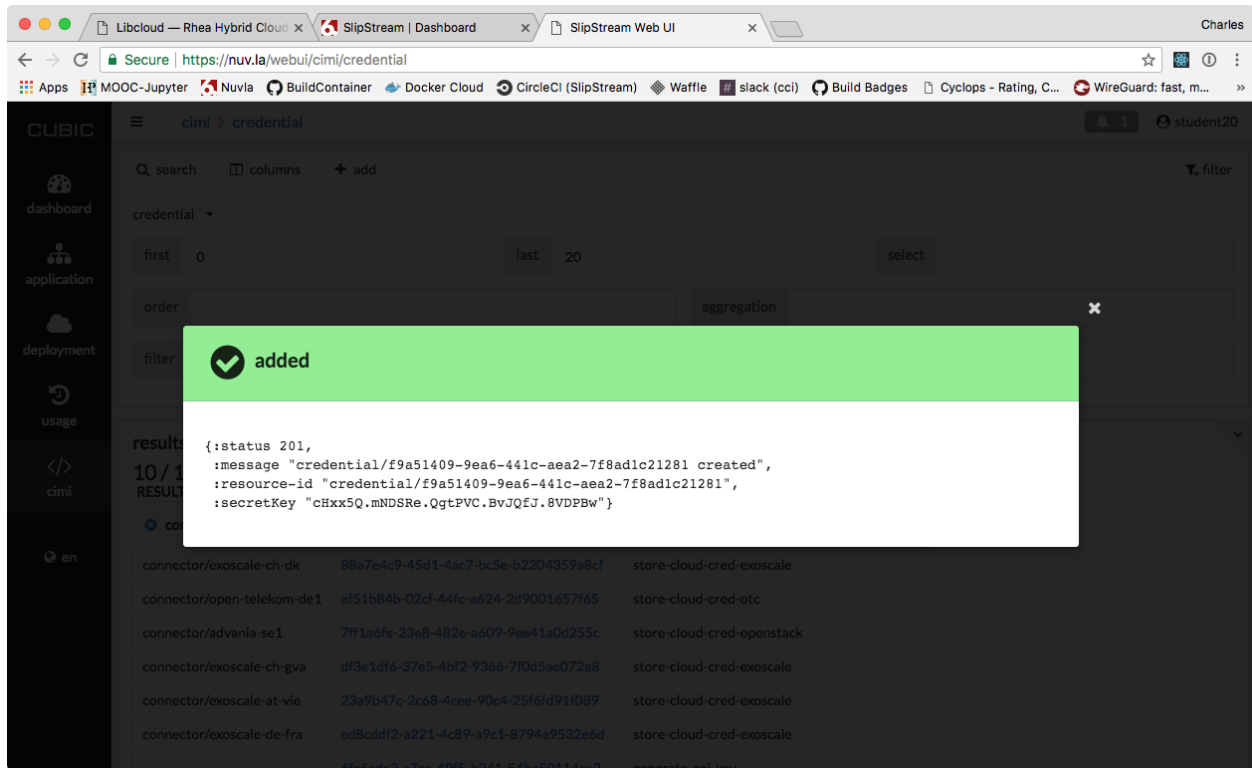
Generating API Key/Secret on Nuvla

We will use the new “CUBIC” interface to Nuvla to generate an API key/secret pair.

- Navigate to the [CIMI credential page](#).
- Click on “search” to show the list of available credentials.
- Click on “add” to create a new credential.
- For the “resource template” select “Generate API key”, if it isn’t already selected.
- Provide optional name and description, if you’d like.
- Leave the TTL at zero.
- Click on the *create* button.

Note: The `ttl` parameter for the API key/secret lifetime (TTL) is optional. If not provided, the credential will not expire (but can still be revoked at anytime.) The TTL value is in seconds, so the above time corresponds to 1 day.

This should bring up a success dialog that looks similar to the following screenshot.



From this dialog, you will need the values of the “`resource-id`” and “`secretKey`” fields.

Configure Terminal with API Key/Secret

Store KEY and SECRET as environment variables in your terminal session.

Copy the secret (`secretKey`) that is returned from the server and export it:

```
$ export SECRET=<...>
```

The *key* is the value of *resource-id* (without the *credential* prefix). Example:

```
$ export KEY=05797630-c1e2-488b-96cd-2e44acc8e286
```

We will use these values when starting a machine via Libcloud with Nuvla.

Using Libcloud for Nuvla deployment

Do the following from the SSH session that you opened on your *Centos-libcloud* machine to start a new virtual machine using your API key/secret credentials.

We will deploy a WordPress instance via the Libcloud API.

- You will need the latest version of the *slipstream-libcloud-driver*:

```
pip install slipstream-libcloud-driver
```

- Open a python session:

```
$ python
```

- Import convenience modules:

```
import os
from pprint import pprint as pp
```

- require modules for the slipstream driver:

```
import slipstream.libcloud.compute_driver
from libcloud.compute.providers import get_driver
```

- create the driver itself:

```
slipstream_driver = get_driver('slipstream')
```

- Log into Nuvla using API key and secret:

```
# KEY and SECRET are taken from the environment

ss = slipstream_driver(os.environ["KEY"],
                       os.environ["SECRET"],
                       ex_login_method='api-key')
```

- Optionally check you can list available images from App Store:

```
pp(ss.list_images(ex_path='examples/images'))
```

- Complete application (node) deployment (WordPress server):

```
# Get the WordPress image
image = ss.get_image('apps/WordPress/wordpress')
```

- Set WordPress Title. You may want to change this to be convinced it is your instance:

```
wordpress_title = 'WordPress deployed by SlipStream through Libcloud'
```

- Create the dict of parameters to (re)define:

```
parameters = dict(wordpress_title=wordpress_title)
```

- Create the Node. After this the node should also be visible in the web browser interface:

```
node = ss.create_node(image=image, ex_parameters=parameters)
```

- Wait the node to be ready:

```
ss.ex_wait_node_in_state(node)
```

- Update the node:

```
node = ss.ex_get_node(node.id)
```

- Print the WordPress URL. Visit the URL to ensure that the service is accessible:

```
print node.extra.get('service_url')
```

- Destroy the node (i.e terminate a deployment):

```
ss.destroy_node(node)
```

Using Libcloud Directly on Exoscale

One of the benefits of the Libcloud API is that the same code can be reused for different cloud providers. Here we will use the same process to deploy on Exoscale.

- Open a python session:

```
$ python
```

- Import convenience modules:

```
import os
from pprint import pprint as pp
```

- Require module for the driver:

```
from libcloud.compute.providers import get_driver
```

- Set variables for expected deployment:

```
location_name = 'ch-gva-2'
image_name = 'Linux CentOS 7.4 64-bit 10G Disk (2018-01-08-d617dd)'
size_name = 'Micro'
deployment_name='libcloud-example'
```

- Set your Exoscale Key and Secret. **Note that these are NOT the same key and secret that you used for Nuvla.** Normally, you'd use the [SlipStream Exoscale instructions](#) to find the correct values. However, your instructor will give you the correct values for the training. Set these variables in the shell:

```
key=...
secret=...
```

- create the driver:

```
exoscale_driver = get_driver('exoscale')
```

- Log into Exoscale using API key and secret:

```
exo = exoscale_driver(key,secret)
```

- Get location:

```
locations = {l.name: l for l in exo.list_locations()}
location = locations.get(location_name)
```

- Get image:


```
images = {i.extra['displaytext']: i for i in exo.list_images(location=location)}
image = images.get(image_name)
```

- Specify expected size:

```
sizes = {s.name: s for s in exo.list_sizes()}
size = sizes.get(size_name)
```

- Deploy the node:

```
# Last parameter is optional, but is set here to allow SSH connectivity to the
↳instance
node = exo.create_node(name=deployment_name, size=size, image=image,
↳location=location, ex_security_groups=['slipstream_managed'] )
```

- Display some results:

```
pp(node)
pp(node.public_ips)
pp(node.extra['password'])
```

- Display help message for SSH connection to the running instance. Log into the machine to ensure that it is working:

```
msg = """ SSH command :
$ ssh centos@{}
# NB : password is {}"""

print msg.format(node.public_ips[0], node.extra['password'])
```

- Destroy the node (i.e terminate the deployment):

```
exo.destroy_node(node)
```

9.1.4 Managing Data with Onedata

Managing Files via Web Interface

All files in Onedata are organized in spaces. The Web User interface allows for uploading and downloading files, creating access tokens, managing access rights, sharing spaces, and joining other users' spaces.

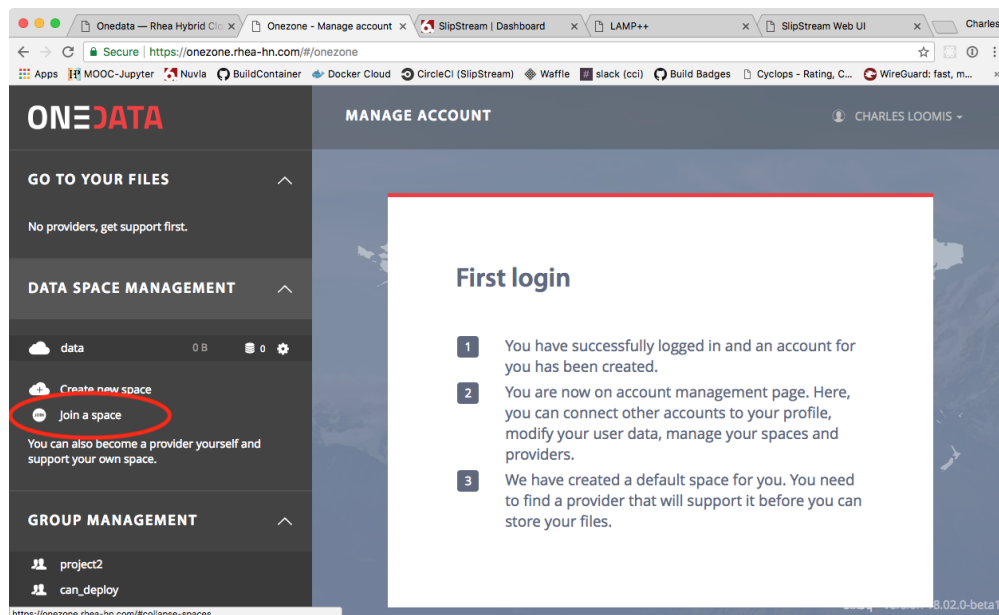
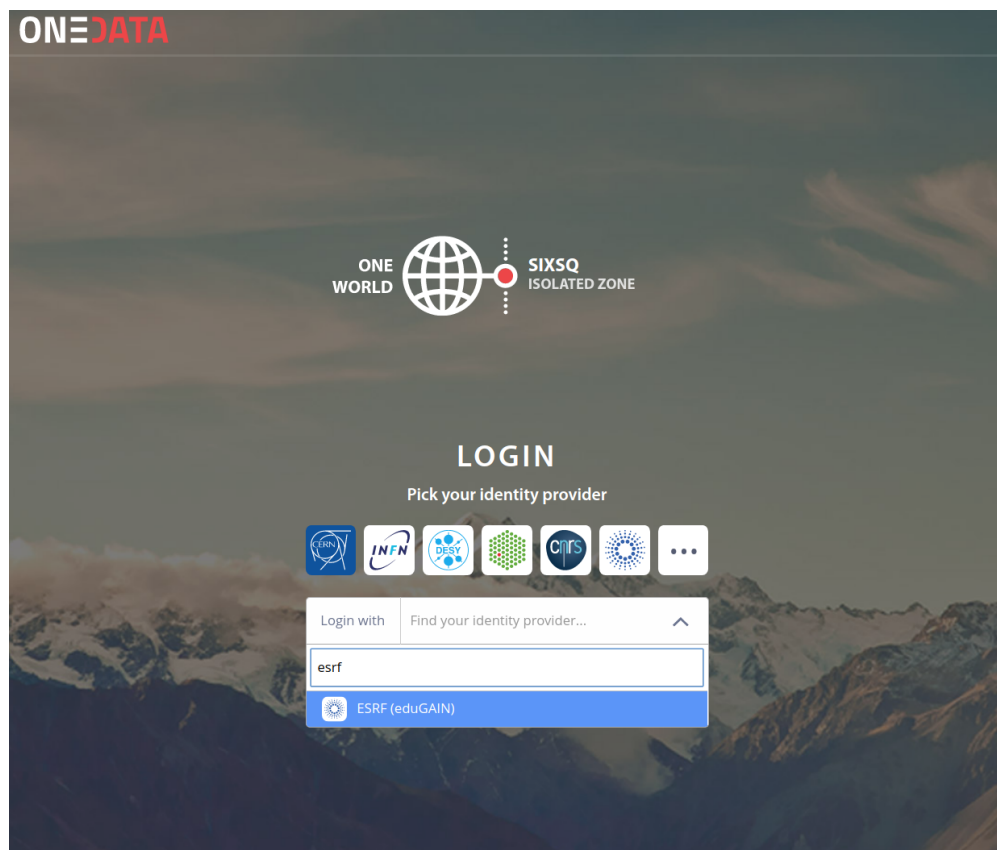
Login to Onezone

The Onezone web interface is already deployed at <https://onezone.rhea-hn.com>. You will use your **standard federated identity** to log into the service, **not your student identity**.

Select Your Federated Login

Join a Space

A training space has been created called “space-load”. In the lefthand menu under the “DATA SPACE MANAGEMENT” section, click on the “join a space” link.



To join the space, you will need a token. A token has been generated for you and associated with your student username. You can find your token in the credentials file you have been given. Copy the token and paste it into the form that appeared.

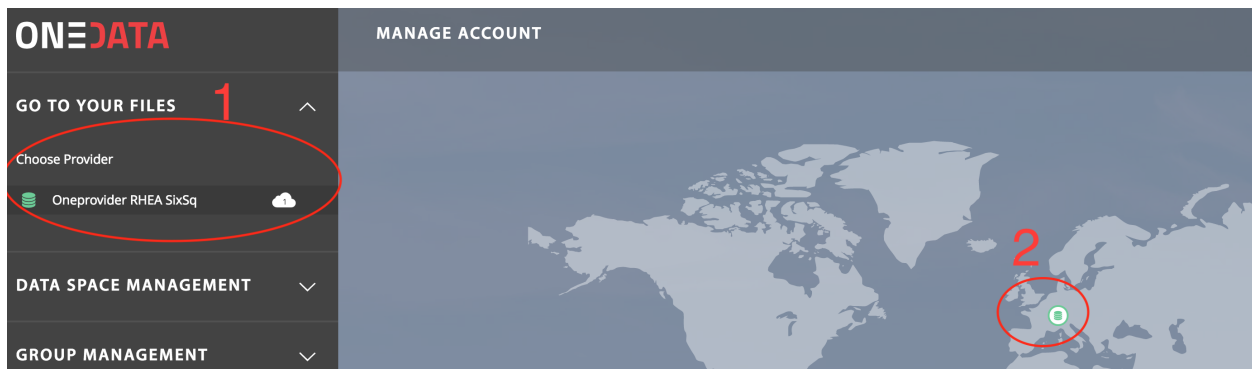
Join a space

Enter a token of a space to join:

Join

Select Your Storage Provider

Onedata enables access to federated storage resources via distributed Oneprovider services, deployed close to actual storage resources in order to enable efficient local access and replication when necessary between the sites. Try to always to connect to the Oneprovider instance, which hosts the data on the storage which is closest to where the computation will be performed.



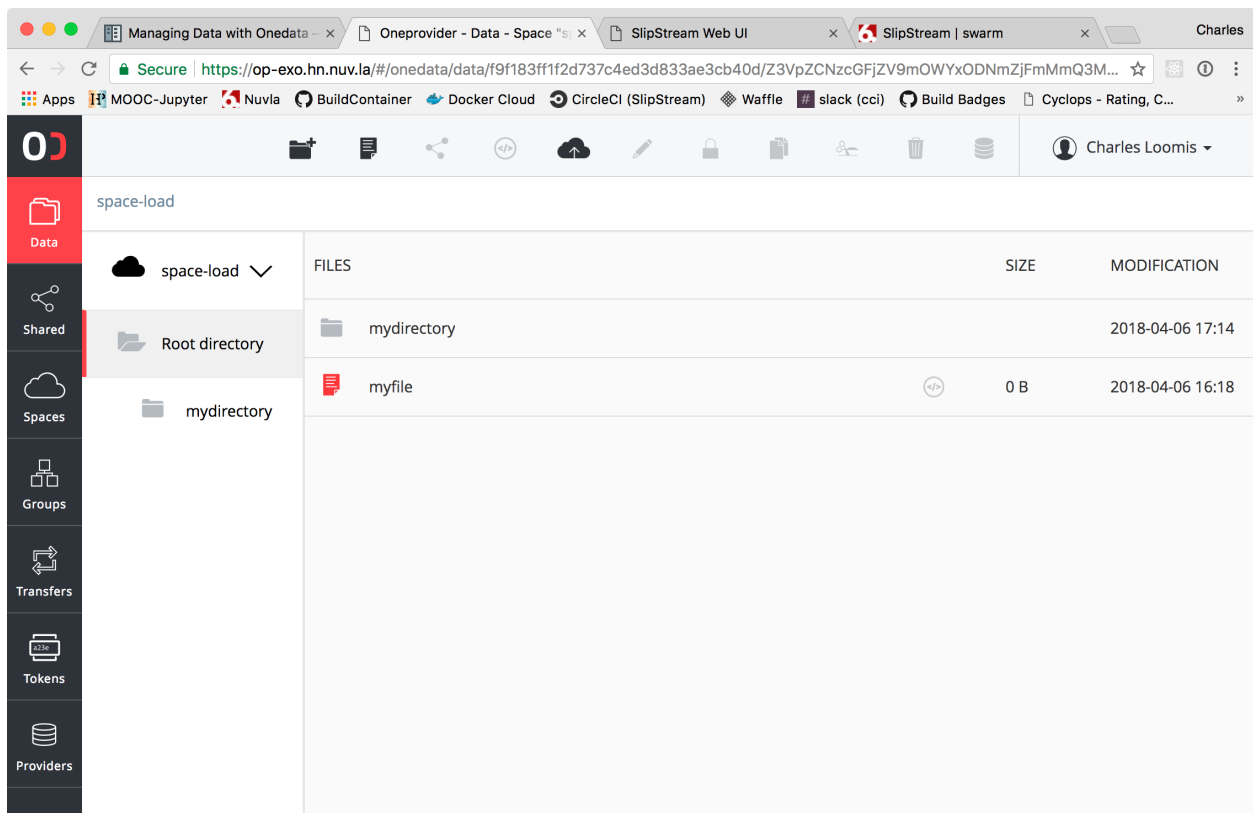
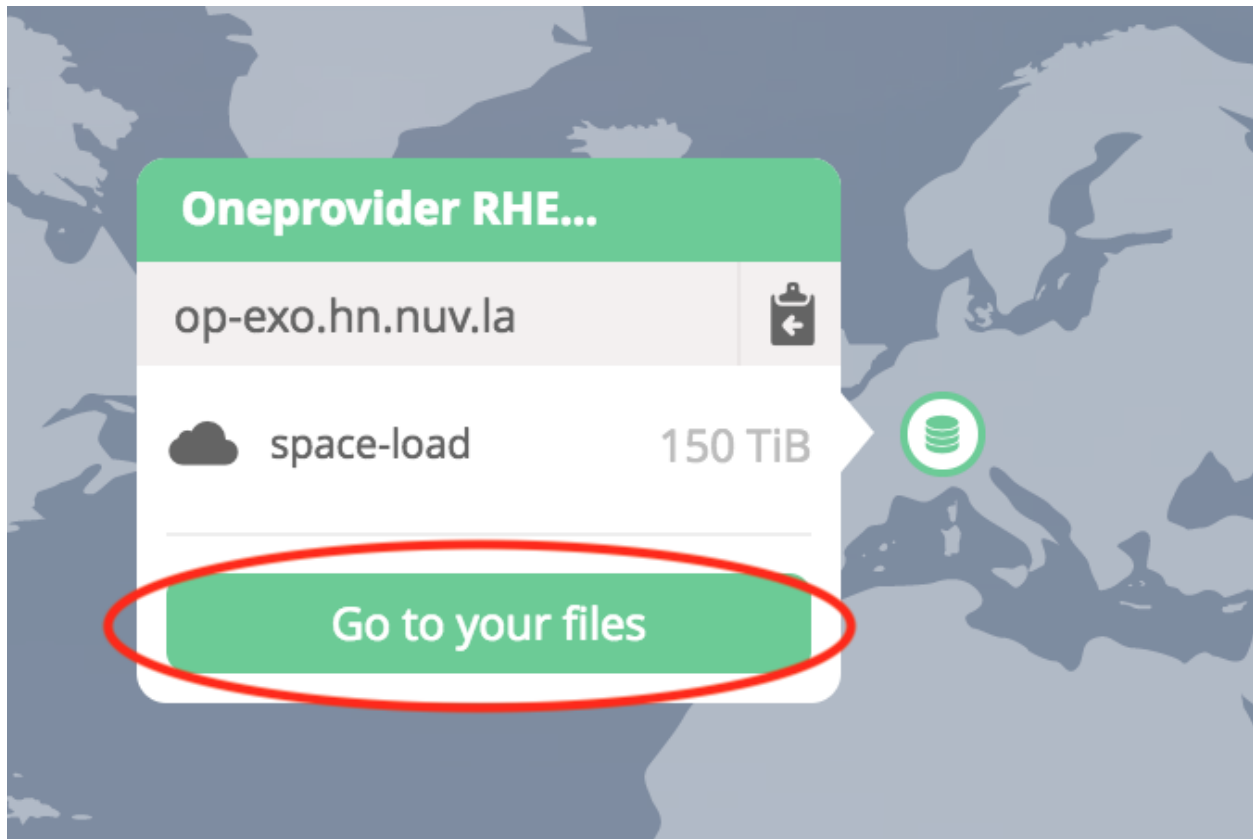
Press **Go to your files** button in the popup. The Oneprovider hostname is displayed in the popup, along with the provider name and storage quota dedicated to this space:

You will be redirected to the Oneprovider page, where the URL starts with <https://op-exo.hn.nuv.la>.

Adding or Modifying Files

On the Oneprovider interface:

- Click on the “Data” tab on the left.
- Select the “space-load” space in the dropdown list of spaces.



You should then see an interface similar to the following screenshot.

This panel will allow you to manipulate files in the space from the browser.

- You can upload a file to a folder, by opening that folder and then dragging and dropping the file into the browser window.
- You can also upload a file by selecting a file after clicking on the “upload” icon.
- Opening (usually downloading) a file simply requires double clicking on the file.
- You can also create directories or files via the associated icons.

From the web interface,

- Create a directory with a unique name.
- Add one or more files to this directory.
- Download one and ensure that it has the correct contents.

We will next verify that these files can be accessed from a virtual machine with Oneclient.

Access Files on a VM via POSIX

Files can also be accessed directly via the POSIX protocol from a Virtual Machine (or another client). Running the Oneclient process provides this functionality.

Create an Access Token

Oneclient requires an access token to interact with the data stored in a space. You can generate such a token from the Onezone service.

If you’re currently viewing the Oneprovider interface, you can return to Onezone by clicking on the “Providers” link in the left menu.

On the Onezone interface, you should see a menu that resembles the following screenshot. Open the “ACCESS TOKENS” section, if necessary.

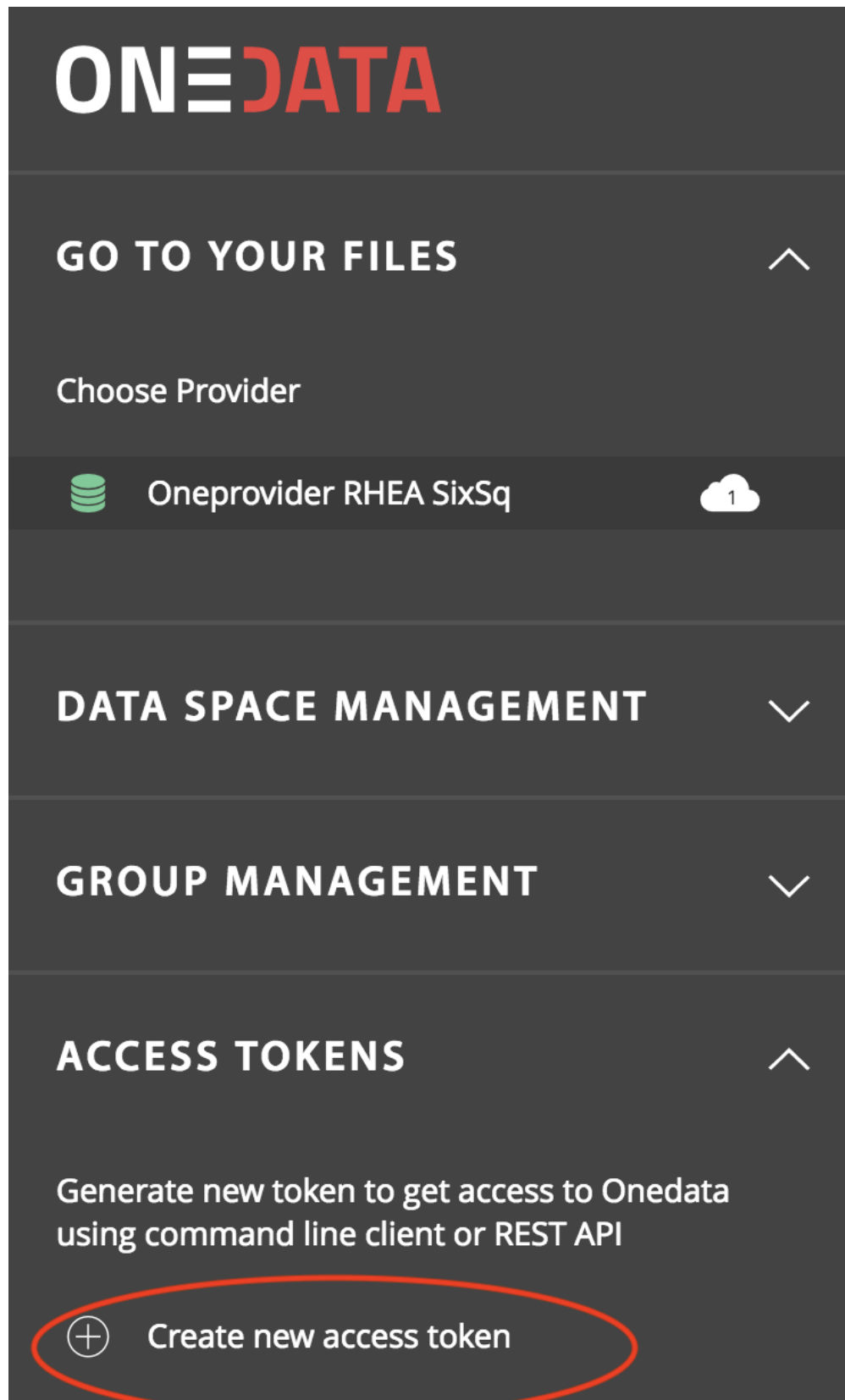
Click on the “Create new access token” action to create a new token. The token will be needed to access spaces via Oneclient.

Deploying a OneClient Application

From Nuvla, go to the page containing the [Oneclient component](#). This will run Oneclient on an Ubuntu 16.04 virtual machine. The component expects four input parameters.

1. **access-token:** Paste the access token you had created in Onezone (see [Create an Access Token](#)) here.
2. **mount point:** The location where data coming from the spaces will be mounted. You can keep the default value.
3. **provider-hostname:** The Endpoint URL of your Oneprovider instance. Here, use the value: `op-exo.hn.nuv.la`.
4. **version:** The Oneclient software version. Use the default value.

As usual, you can optionally set tags for the deployment. When you’ve provided all of the information, **click the “Deploy Application Component” button**.



Deploy Application Component

Global parameters

SSH access

☒

?

Cloud

exoscale-ch-gva* - € 0.0215/h (2/2048/10 Small linux) [CH]

?

Tags

?

Input parameters

More ▾

Input

access-token

?

Input

mount-point

/mnt/onedata

?

Input

provider-hostname

?

Input

version

18.02.0.beta1-1

?

Deploy the application component `HNSciCloud/onedata/oneclient-ubuntu16.04` (v16904) on the selected cloud service.

Cancel

Deploy Application Component

SSH Connection to the VM

1. From the Nuvla dashboard, wait for the OneClient deployment to be ready.
2. Log into the virtual machine via SSH either by clicking on the *Service URL* for the deployment or by logging in manually. The username for this image is “root”.

Browse Onedata Files

1. Browse the directory which was set as mount point for Onedata. If you didn’t change the default it will be `/mnt/onedata`.

In this directory, you should find a `space-load` folder which corresponds to the `space-load` space in Onedata.

2. Read files from Oneclient:

```
$ ls -lh /mnt/onedata/space-load/
```

It should contain the files you uploaded from the web interface. There will likely be other files there because the other students are also using the same space.

3. Creating files from Oneclient

Either:

```
$ touch /mnt/onedata/space-load/studentXX-test.txt
```

Or:

```
$ echo Grenoble > /mnt/onedata/space-load/studentXX-test.txt
```

After refreshing your Oneprovider web page, you should see your new files in the web interface as well.

9.1.5 Conclusions

This training will covered three activities:

1. Deploying cloud applications via Nuvla,
2. Taking advantage of the Libcloud API for multi-cloud management, and
3. using Onedata services for managing data.

Further information can be found in the [platform documentation](#) particularly the “Researchers” section. For all support requests, please send an email to support@sixsq.com.

9.2 Procurers Training

Because of the limited time available, this training is presented as a demonstration rather than a hands-on tutorial. This demo covers three topics:

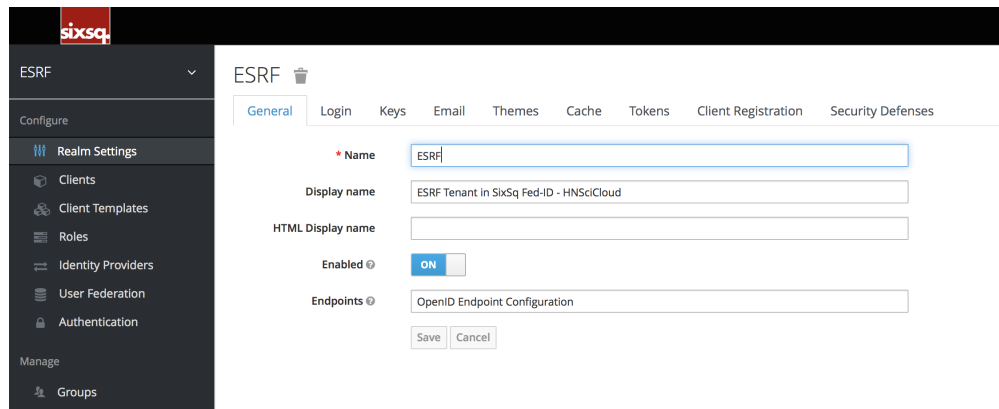
1. User Management in Keycloak
2. Getting Usage Information
3. Adding Storage to Onedata

Further documentation can be found in the rest of the [platform documentation](#), particularly the “Administrators” and “Data Coordinators” sections.

9.2.1 KeyCloak Administration

The KeyCloak server for the RHEA Consortium allows you to manage the users (and their rights) in your tenant. All the Buyers Group organizations have at least one tenant administrator with rights to make changes in Keycloak.

The KeyCloak server can be found at the address <https://fed-id.nuv.la>.



Keycloak is extremely flexible with respect to the user management processes. We have created some simple workflows, but would like to refine them to make them more relevant for the Buyers Group organizations.

The workflow for accepting users to your tenant You can accept users from any identity provider connected to eduGAIN or Elixir AAI. You can also create “local” users on Keycloak for people who do not have credentials for an identity provider connected to one of those federations.

Default Policies

By default, any authenticated user from eduGAIN or Elixir AAI can log into Nuvla/Onedata using any tenant they choose. The tenant administrator can set various policies to change who can access their tenant and what rights users have.

Users

When you have logged into Keycloak as an administrator, you can see an overview of your users by clicking on *Users* in the lefthand menu bar.

This menu option brings you to the user list page. In the search box you can type in a full name, last name, or email address you want to search for in the user database.

For a particular user, you can:

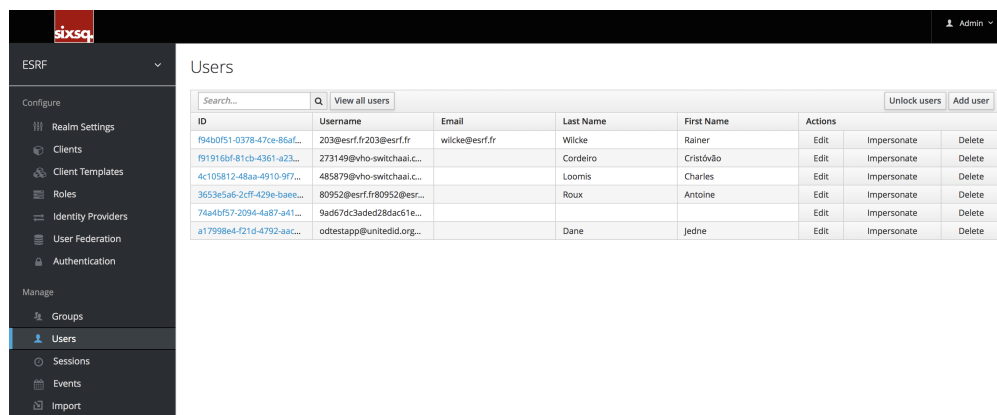
- Enable or disable the user.
- See role and group membership.
- View any active sessions.

Changes you make, take effect **the next time the user logs in through Keycloak**. Note that session cookies, cached sessions in Keycloak, cached session in the Identity Providers, etc. may affect when changes really take effect.

Add a Local User

Most users will authenticate via an external identity provider. However, you may need to create a “local” user for someone who does not have credentials for an identity provider in the eduGAIN or Elixir AAI identity federations. “Local” means a user created directly in Keycloak.

On the right side of the user list, you should see an *Add User* button. Click that to start creating your new user.



When viewing a user, you can manage their credentials on the “Credentials” tab. You can also assign groups and roles via separate tabs.

Blacklisting and Whitelisting

We provide two simple policies for limiting access to your tenant: white and black lists. Although crude, they are effective for limiting access to your tenant.

See *Blacklisting Users* and *Whitelisting Users*

Blocking New Users

This is useful for allowing existing users to continue using the system, but block any new users.

See *Blocking New User Logins*

Managing Groups and Roles

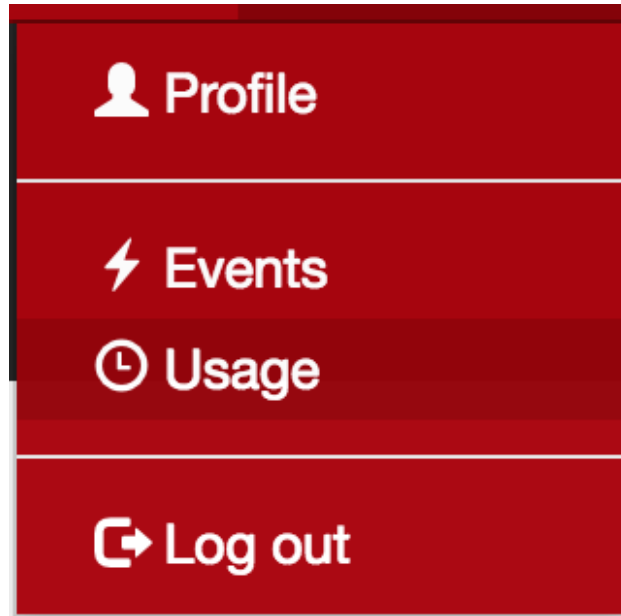
Groups and roles defined within Keycloak are transitted to Nuvla and Onedata and can be used for authorization decisions.

See *Manage Groups* and *Managing Roles*

9.2.2 Usage monitoring in Nuvla

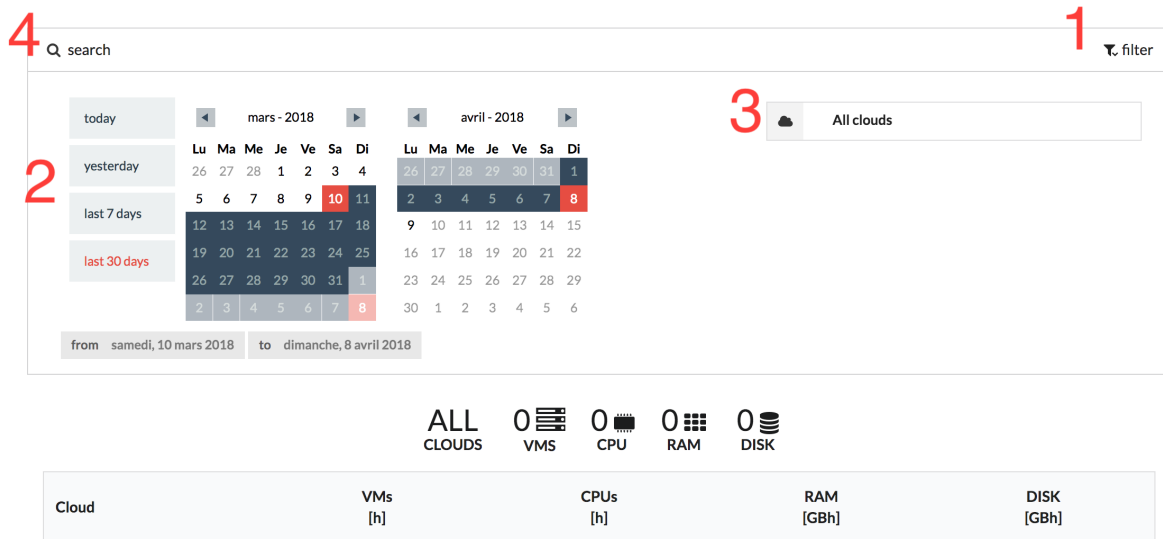
Resource Monitoring

In the standard Nuvla browser interface, you can open the usage page by clicking *Usage* under the profile menu at the top. You can also see [this page](#) in the new browser interface, CUBIC.



All the resources being used, that are visible from a defined credential in Nuvla, can be seen here. This includes resources started directly on the underlying clouds.

Usage Information



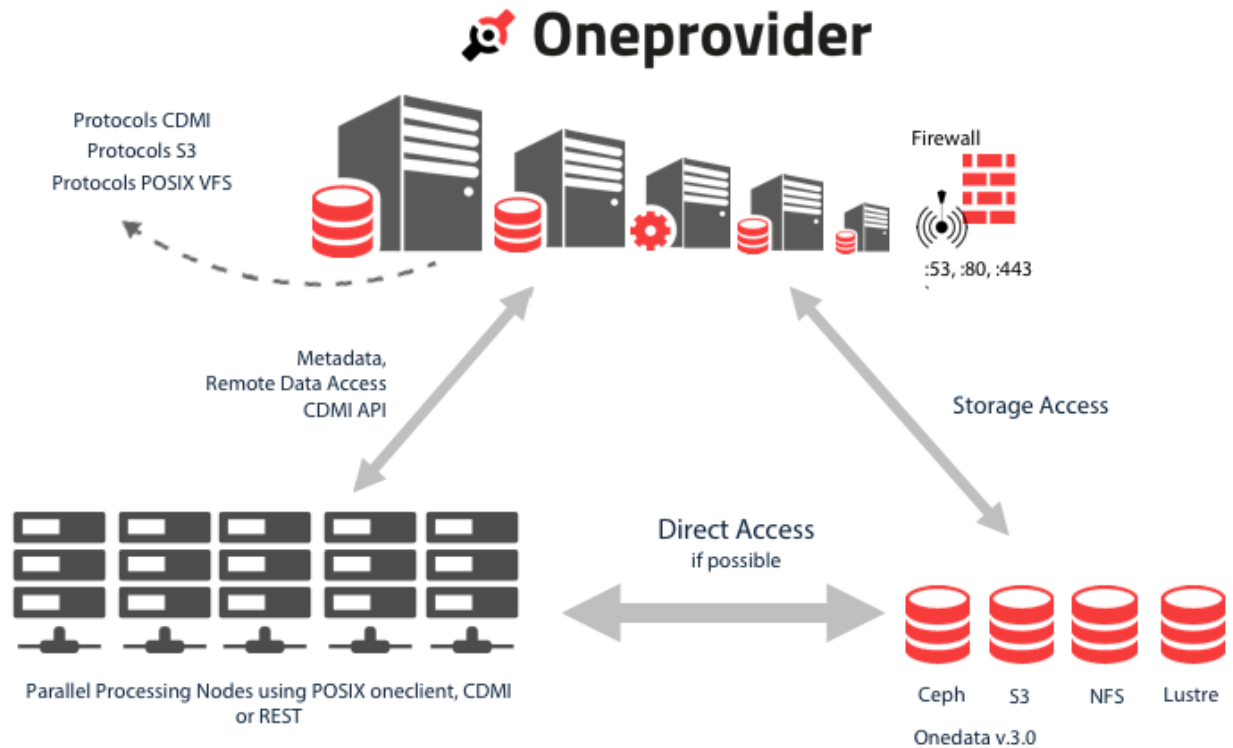
1. Click the filter icon to select parameters
2. Choose a period in the calendar, either a custom or predefined one.
3. Choose cloud provider(s).
4. Update the results with the search button.

Users will be able to see their own usage. Group administrators or SlipStream administrators will be able to see more information.

Live Statistics

You can check that the information is live by deploying a cloud application and checking that the usage grows. For a demonstration, I'll deploy a large Docker Swarm to see that the value grows with time.

9.2.3 Onedata Management



Oneprovider implements drivers for storages such as NFS, GlusterFS, Ceph, Openstack SWIFT and S3.

In order to function properly, Oneprovider needs to register with a Onezone service, which requires a public IP address and specific ports opened to the world (see [Firewall setup](#)).

We have set up a sample Onezone deployment at <https://onezone.rhea-hn.com>.

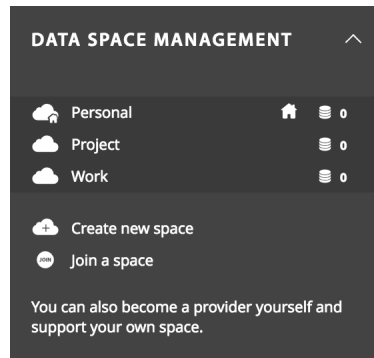
Once logged in, we will add a new space and later support it with a storage from the Oneprovider instance.

Create a Space

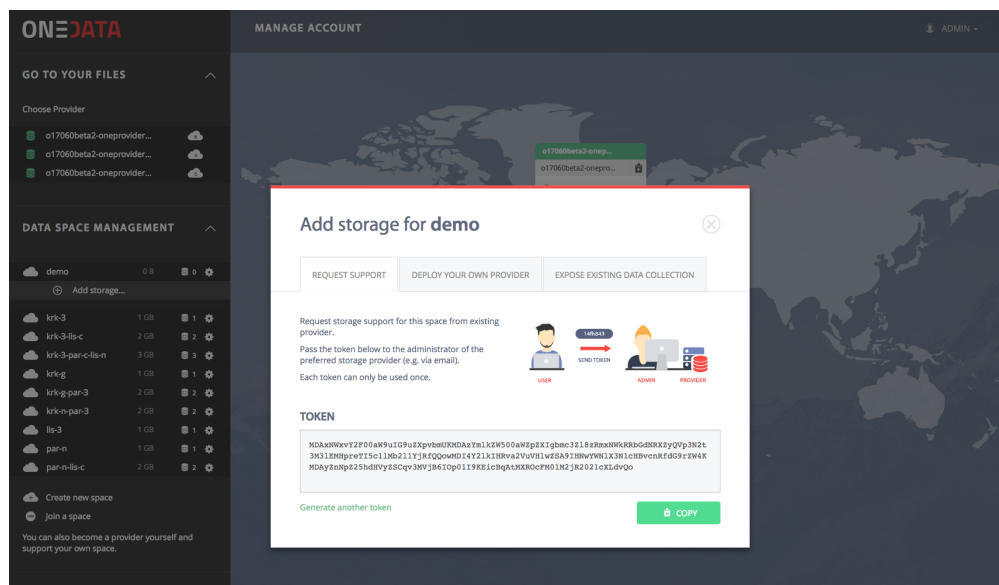
1. In the Onezone Web Interface unfold Data space management tab located on the left menubar.
2. Click Create new space button.
3. Provide new space name in the text edit field and confirm.

Create Space Support Token

1. Click on the newly created space on the left to unfold options



2. Click *Add storage* button
3. Copy the generated token from the popup window



Add Storage to the Oneprovider

1. Open Oneprovider Onepanel administration interface: <https://op-exo.hn.nuv.la:9443>
2. Login as user *admin* with the provided password
3. Click on *Storage* tab on the left
4. Press *Add storage* button in the top right
5. Select type of storage *S3*
6. Fill in the provided S3 bucket credentials
7. Set *LUMA enabled* radio button to Off
8. Set *Signature* version to 4
9. Set *Read only* button to Off
10. Set *Insecure* radio button to On
11. Press *Add* button

The screenshot shows the 'STORAGES' configuration page in the HNSciCloud Clusters interface. The left sidebar includes a search bar and navigation tabs for 'Nodes', 'Provider', 'Storages' (which is highlighted), and 'Spaces'. The main content area is titled 'ADD A NEW STORAGE' and contains the following fields and controls:

- Storage type:** A dropdown menu set to 'S3'.
- Storage name:** A text input field containing 'MyS3Bucket' with a green checkmark.
- LUMA enabled:** A toggle switch that is currently turned off.
- Admin access key:** A text input field containing 'accessKey1' with a green checkmark.
- Admin secret key:** A text input field with masked characters (dots) and a green checkmark.
- Hostname:** A text input field containing 's3.amazonaws.com' with a green checkmark.
- Bucket name:** A text input field containing 'MyBucket' with a green checkmark.
- Signature Version:** Radio buttons for '4' (selected) and '2'.
- Block size [bytes] (optional):** An empty text input field.
- Timeout [ms] (optional):** An empty text input field.
- Read only:** A toggle switch that is currently turned off.
- Insecure:** A toggle switch that is currently turned off.
- Add:** A green button at the bottom right to submit the configuration.

Support the User Space

1. Click on the *Spaces* tab on the left
2. Select storage name to the one assigned in the previous step
3. Paste the support token created in the Onezone to *Support token* field
4. Select support size and units - this will be the maximum amount of data occupied by this space on this storage
5. Set *Mount in root* radio button to Off
6. Set *Import storage data* radio button to Off
7. Press *Support space*

Check the Space Support Settings

1. Click on *Spaces* tab on the left
2. Select the newly support space from the list

Check the Space Storage Space in Onezone

1. Open Onezone interface in the browser: <https://onezone.rhea-hn.com>, or refresh if necessary
2. Click on the supported space on the left
3. Check the available storage

Clusters

Search...

ERFI-EOSC-Cyfronet

Nodes

Provider

Storages

Spaces

SPACES

Cancel supporting space

ADD SUPPORT FOR A SPACE

Storage: MyS3Storage

Support token: mxYQpUBQtD-al5dEQNB_k3q5VGWjxpgjibkMEdtTNphasdnl

Size: 1

☒ MiB
☐ GiB
☐ TiB

Mount in root: ☐

Import storage data: ☐

Support space

CHAPTER 10

Consortium

The consortium consists of six organizations that have experience working together to deliver computing solutions. Rhea leads the consortium consisting of software providers (SixSq and CYFRONET) and cloud service providers (Advania, Exoscale, and T-Systems).



Experienced leader of large frame contracts in the Space and Defence sectors, including software centric systems and software products. Majority shareholder of SixSq, RHEA has developed a solid relationship with the cloud specialist and supports spin-in of its technologies into the space and defence sectors. RHEA acts as the prime for the project and performs much of the system testing. Rhea is based in Belgium. [\[more info\]](#)



Responsible for technical coordination, SixSq brings its knowledge of cloud technologies and its innovations from the Nuvla cloud application management platform. SixSq is based in Geneva, Switzerland. [\[more info\]](#)



Provides the Onedata data management solution and support for it. Onedata is seamlessly integrated into the Nuvla/SlipStream solution, allowing for easy deployment of the platform. Academic Computer Centre CYFRONET AGH is based in Krakow, Poland. [\[more info\]](#)

Advania is a leading Nordic IT-provider serving thousands of corporate clients in the public and private sector. Advania provides a cloud infrastructure located in Iceland and optimized for high-performance applications to the Consortium's hybrid cloud platform. [\[more info\]](#)

Exoscale is the cloud computing platform for cloud native teams. Relying only on pure “as-a-service” components, Exoscale is built by DevOps for DevOps. Originally based in Switzerland, it provides cloud resources in Switzerland, Austria, and Germany to the Consortium’s hybrid cloud platform. [\[more info\]](#)

T · · Systems ·

European leader in IT service delivery, including several cloud services, both public and private. Open Telekom Cloud, with resources in Germany, forms the foundation of its contribution to the project. The company’s experience will also be key in contributing to the business model proposed for the resulting service of this project. [\[more info\]](#)