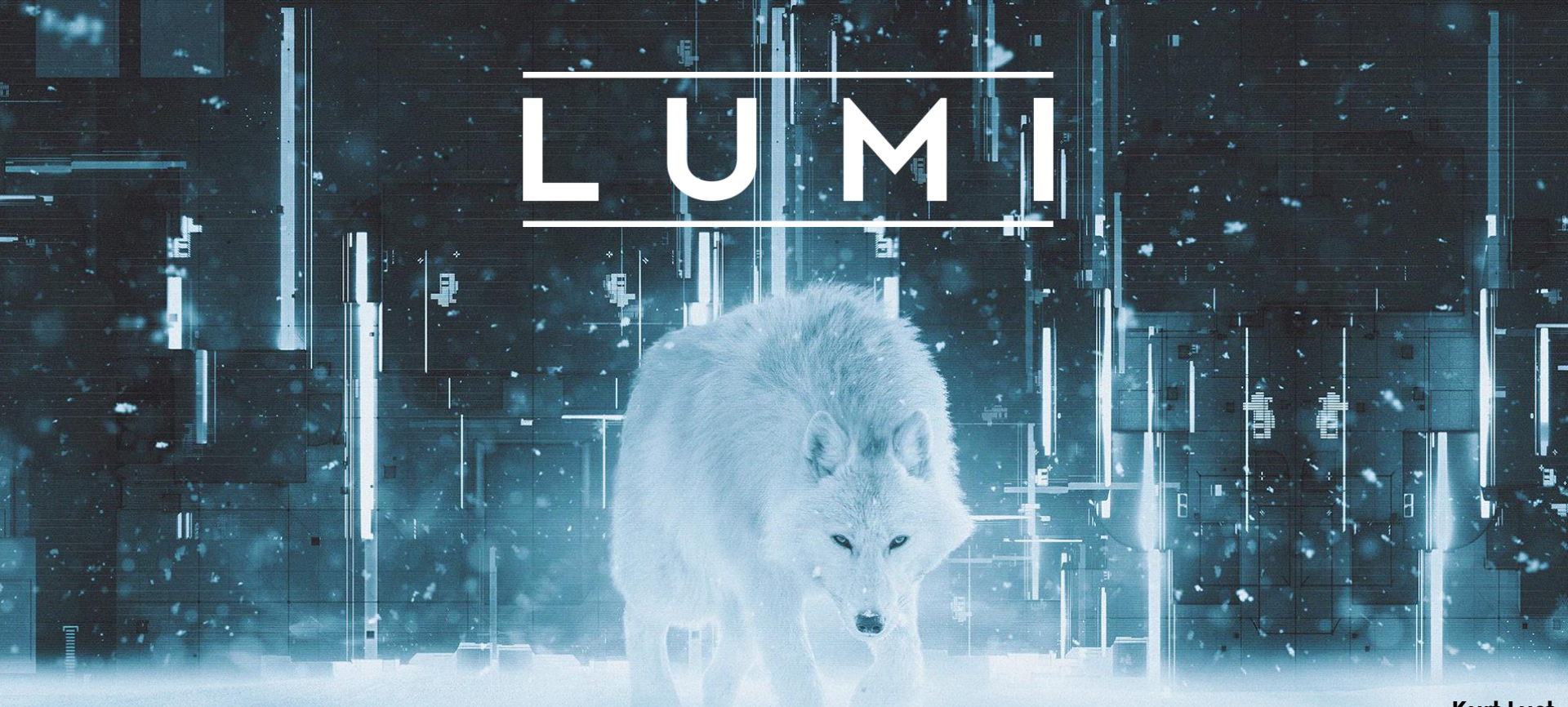


Files on LUMI 2: LUMI-O object storage

Kurt Lust
LUMI User Support Team (LUST)
University of Antwerp

May 2024

Why do I need to know this?

- LUMI-O is the primary option on LUMI to transfer large amounts of data to LUMI
- LUMI-O is the only local option if you want to backup some data
- Some datasets come in a format optimised for object storage rather than a parallel file system

- What we will discus:
  - Properties of object storage
  - Getting started
  - But not a reference manual of the tools that can be used on LUMI-O

LUMI

What is LUMI-O?

- Object storage system, based on Ceph
  - Finnish users: similar to Allas, but less functionality at the moment
  - Specific tools to access data, not mounted as a regular file system
- Organisation:
  - Buckets: "Containers" used to store objects.
    - Flat structure: Buckets cannot contain other buckets
  - Objects: Any type of data, stored in a bucket
    - Atomic access to objects: Put, get, copy, delete, …, but no partial write
  - Metadata for buckets and objects
    - Bucket: e.g., access rights
    - Custom metadata possible

What is LUMI-O? (2)

- Objects can be served on the web also
  - This is how recordings of some LUST courses are served
  - But not meant as a data publishing service (e.g., no EUDAT alternative)
- Can be reached easily from outside LUMI
  - So also a mechanism for data exchange
  - Tools of object storage are more performance and more robust than sftp
- Specs:
  - Capacity: 30 PB
  - Quota: 150 TB capacity, 1K buckets and 500K objects per bucket (fixed)
  - Billed at 0.5 TB·hour per TB per hour
  - Persistent for the duration of the project

Lustre vs LUMI-O (1)

Lustre	LUMI-O object storage
Closely integrated with compute nodes	Separate system
Upgrades with the system	Separate upgrade cycle and on-the-fly
Organisation: Hierachical directory structure and files	Organisation: Double flat space of buckets and objects
Files can be read, written, modified, appended, …	Simple atomic operations on objects: put, get, copy, delete
Optimised for bandwidth to the compute nodes	Optimised for reliability
Simpler schemes for redundancy	Very complex internal redundancy setup

Lustre vs LUMI-O (2)

Lustre	LUMI-O object storage
Integrated in the authentication of the supercomputer	Separate key-based authentication mechanism
Seen as any other POSIX file system	Separate range of access tools/APIs, some tools can provide a filesystem view
No external access	External access integrated, includes web
Structure with MDS and ODS	Structure with MDS and ODS, but very different technologies
Parallelism for performance: Access a file in parallel from multiple processes	Parallelism for performance: Different processes access different objects
Fairly expensive to very expensive hardware	Cheaper hardware

LUMI

LUSTRE vs LUMI-O (3)

- The optimal way/technology of storing data is very different depending on whether you work from a parallel file system or from object storage. E.g., in earth and climate science:
  - netCDF is a popular data storage format for storing simulation data on a parallel filesystem
    - Not suited for object storage though as it would be a single object
  - Zarr is a format to store similar data on object storage (cloud storage)
    - It is not a single object, but a structured collection of objects
    - Putting it on a parallel filesystem where each object would become a file in a directory tree is a very bad idea!
    - But with the right libraries, you can access Zarr data on object storage directly from your application

Accessing LUMI-O

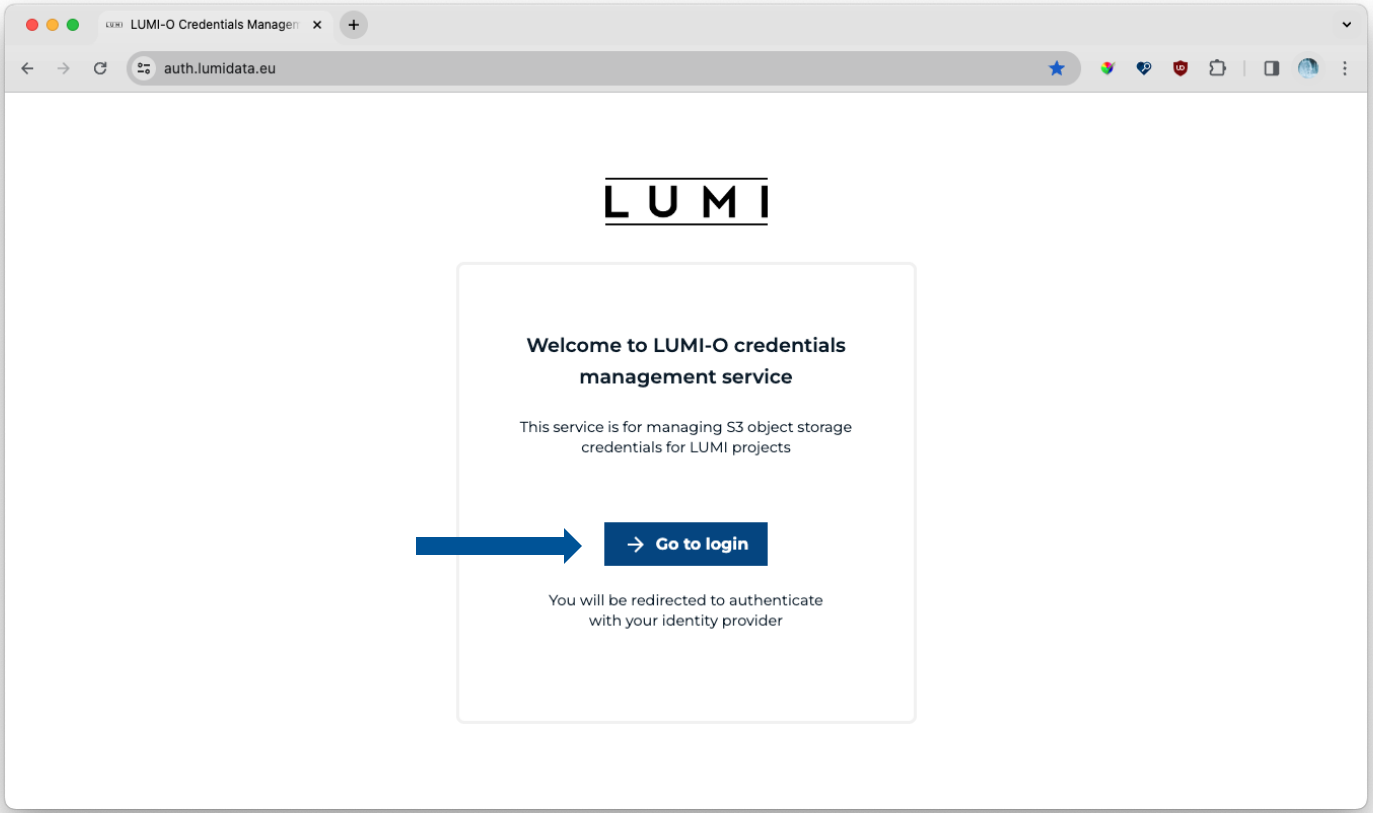
- Access is based on keys
  - Generated via a web interface: Separate steps to generate the credentials and to get them on LUMI
  - Or generated via Open OnDemand: Will put credentials on LUMI for TODO.
- Tools:
  - `rclone`: Easiest tool if you want public and private data
  - `s3cmd`
  - `restic`: More a backup tool

  module `lumio`
  - `boto3`: Python API from the AWS SDK for programmatic access
    - Needs a more recent Python version than the system Python
  - Additional GUI-based tools exist for clients
  - Open OnDemand web interface is not a substitute!
    - Speed limited by browser protocols

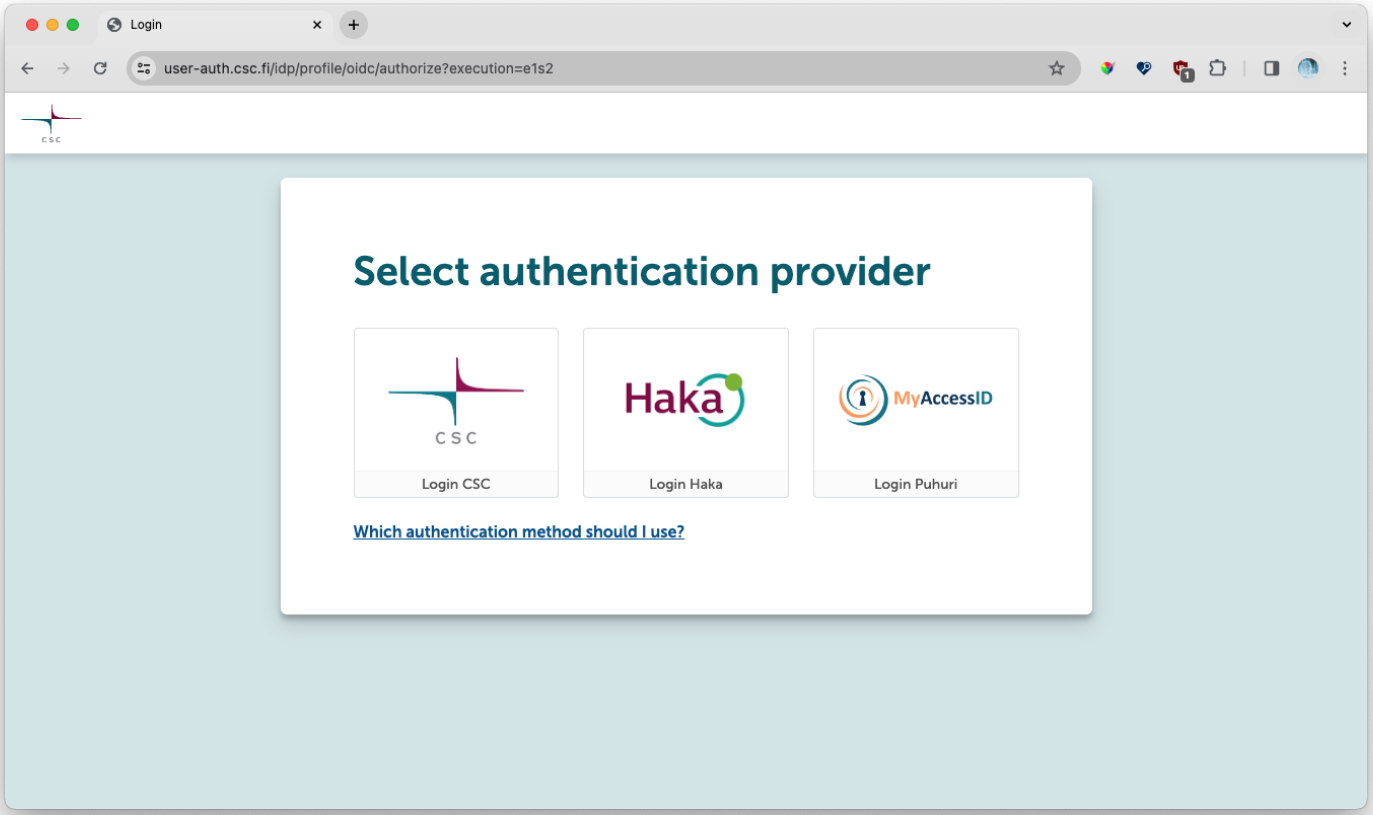
Credential management web interface

- Credential management web interface at auth.lumidata.eu
  - Create keys
  - Extend lifetime of a key
  - Create configure scripts for various tools
- You'll have to select your login method in the same way as for Open OnDemand
- After a while you should see a list of projects, select the one for which you want to generate a key
  - The right column will show active keys for the project, and expired ones
- Selecting an active access key changes the right column to one where you get information about the key, can extend the key and can generate templates to configure various tools

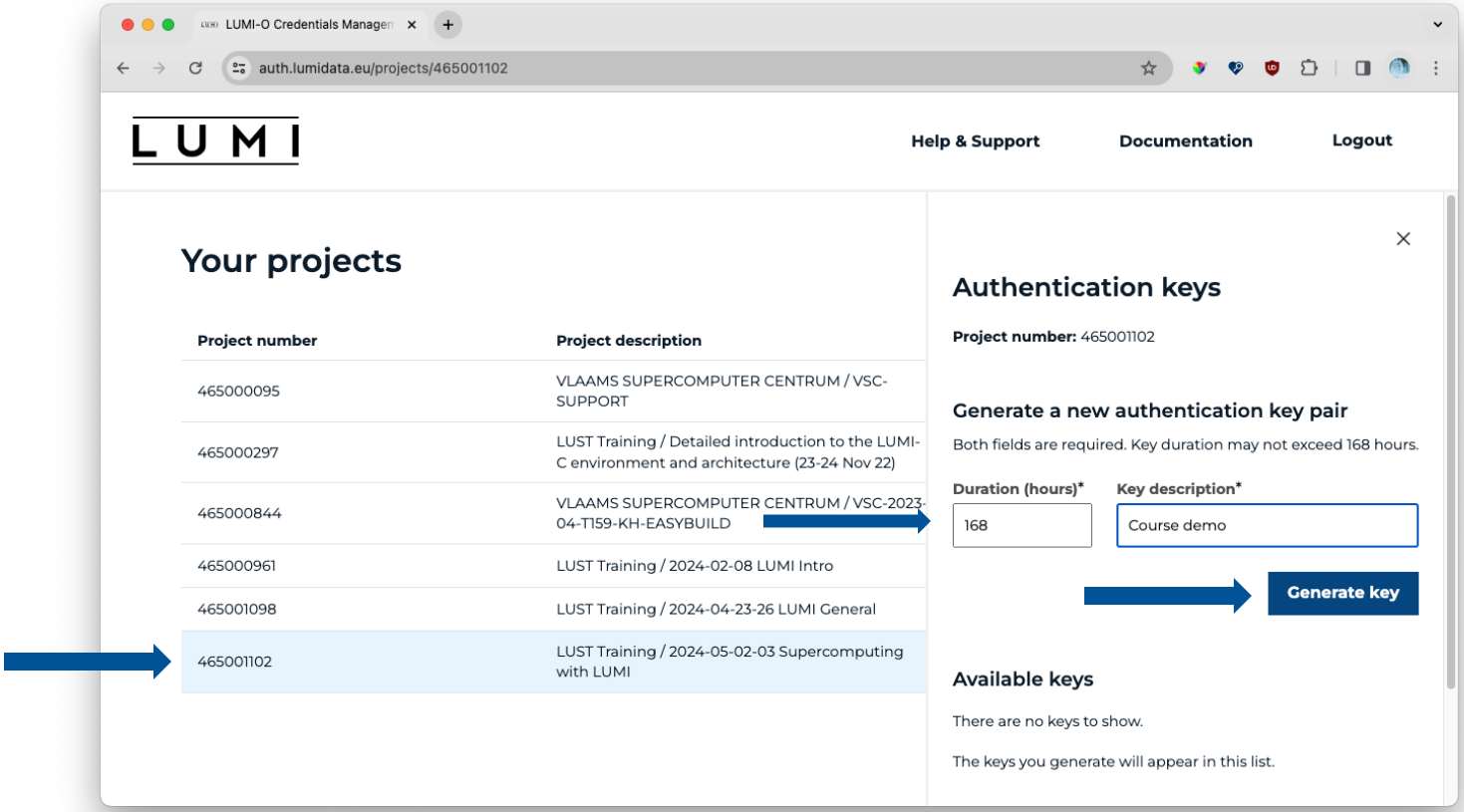
Credential mangement web interface: Create credentials (1)



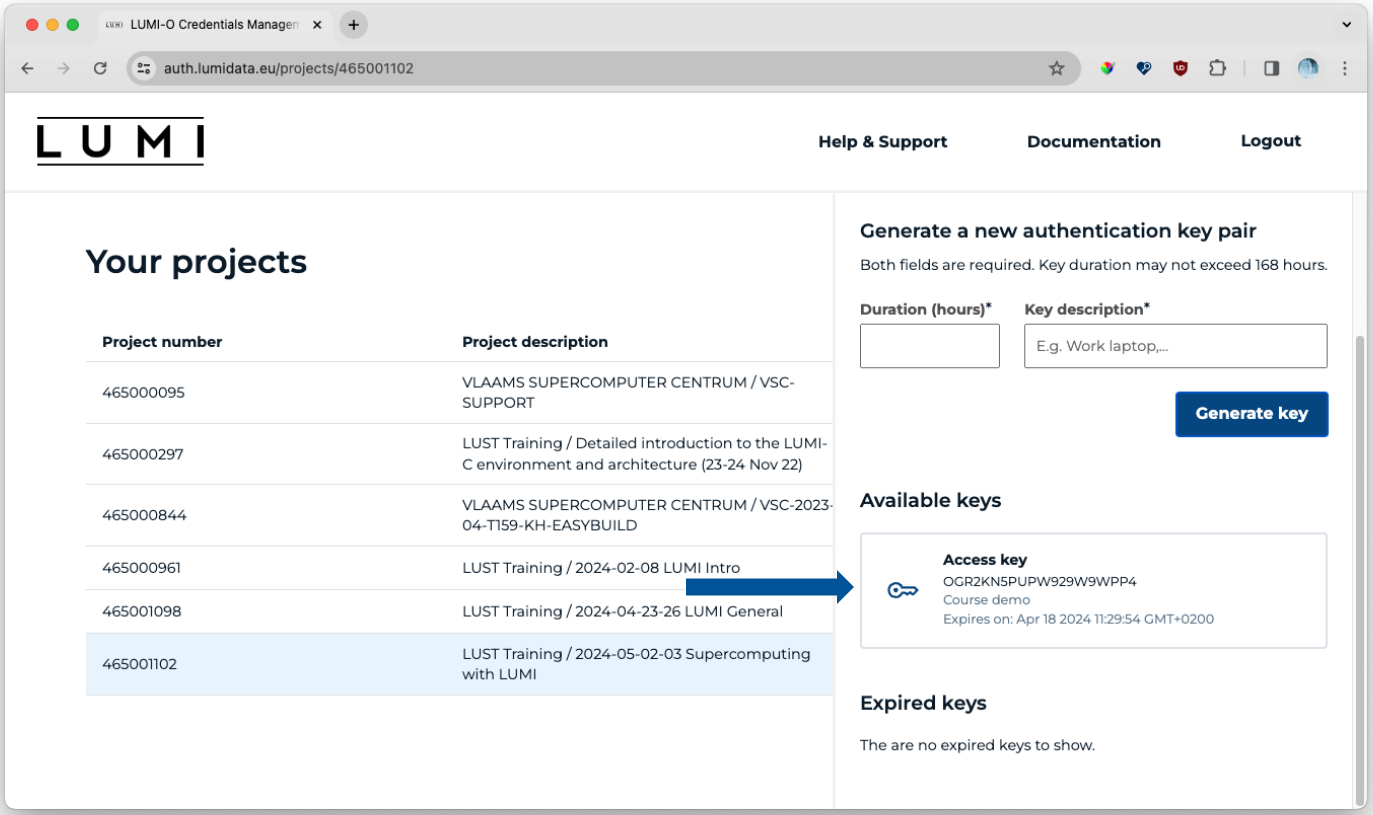
Credential mangement web interface: Create credentials (2)



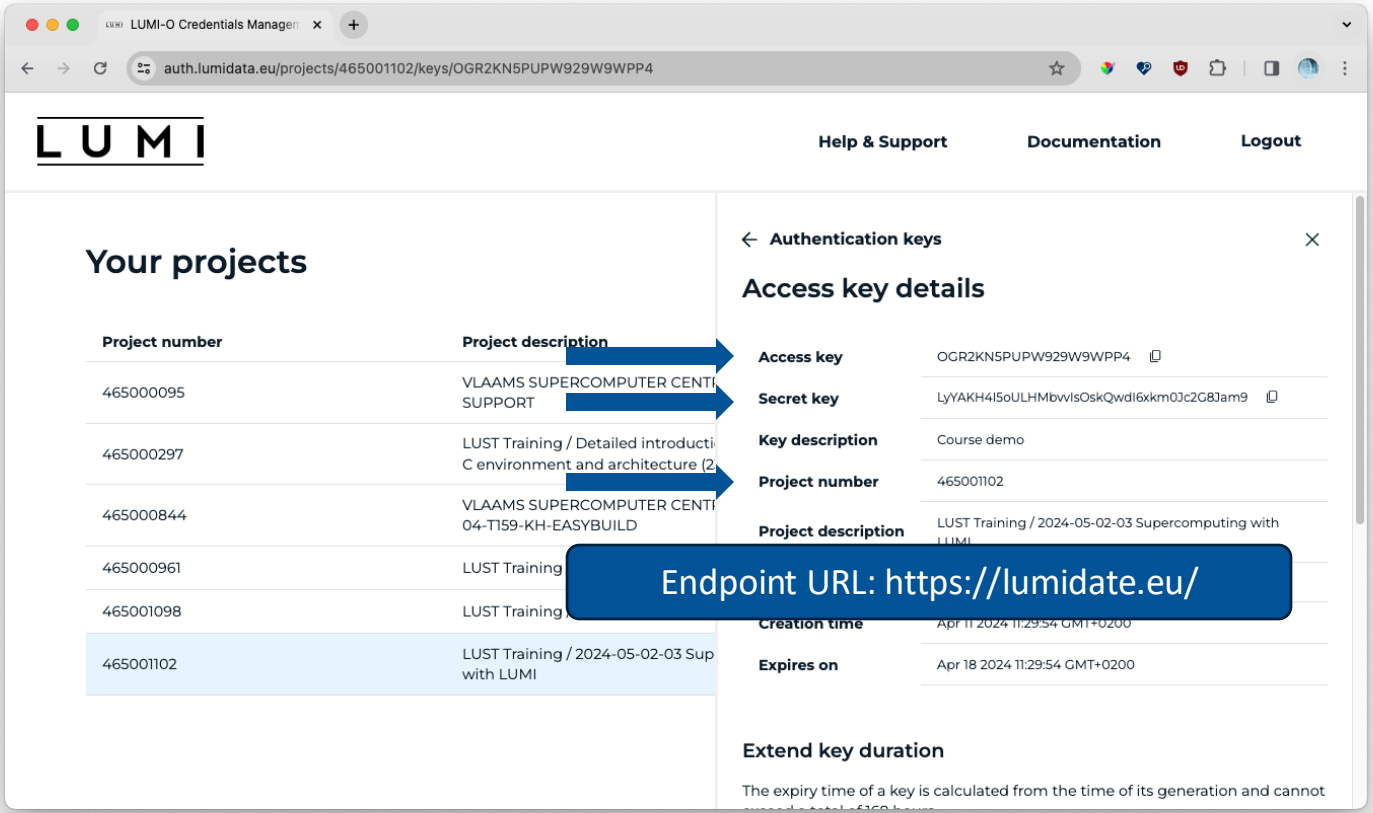
Credential mangement web interface: Create credentials (3)



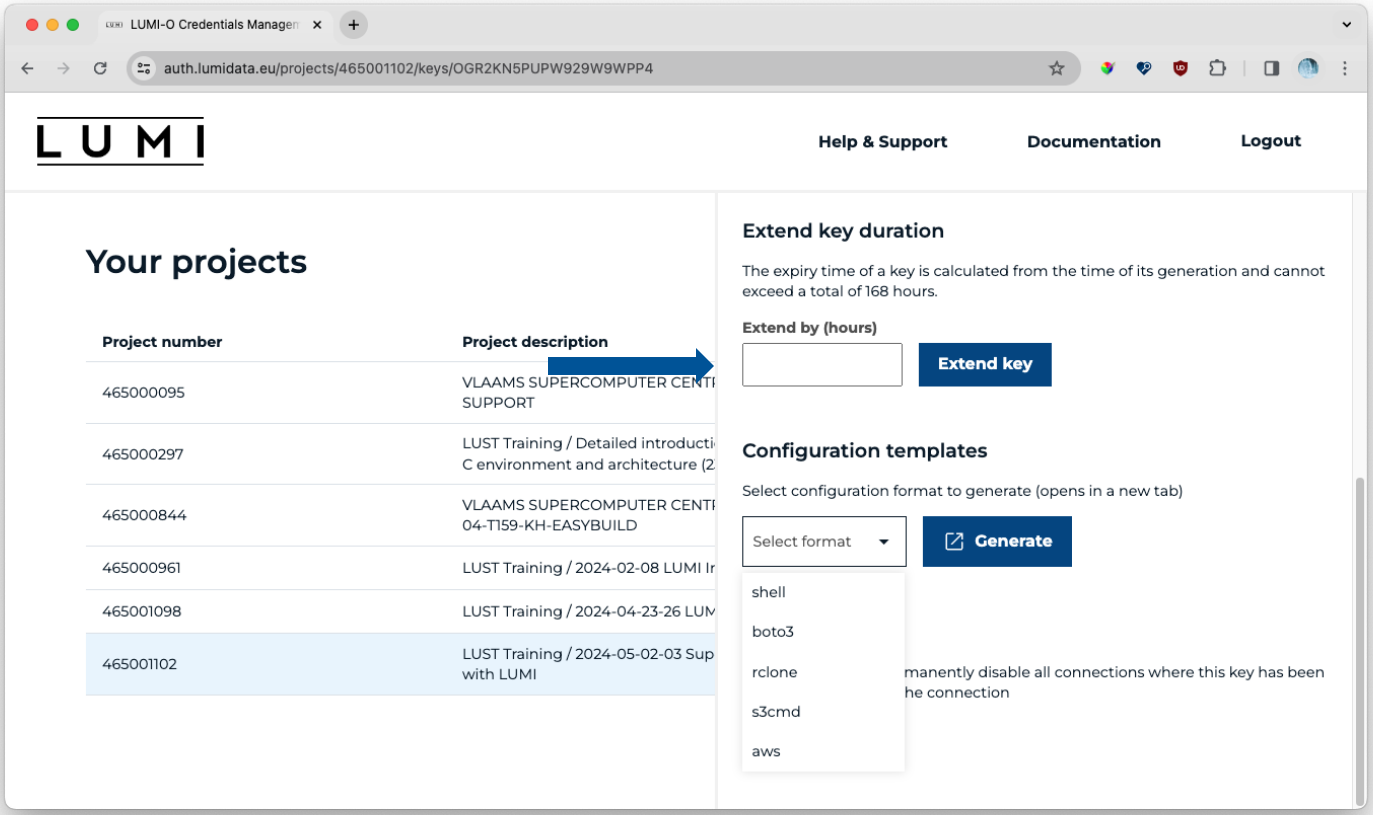
Credential mangement web interface: Create credentials (4)



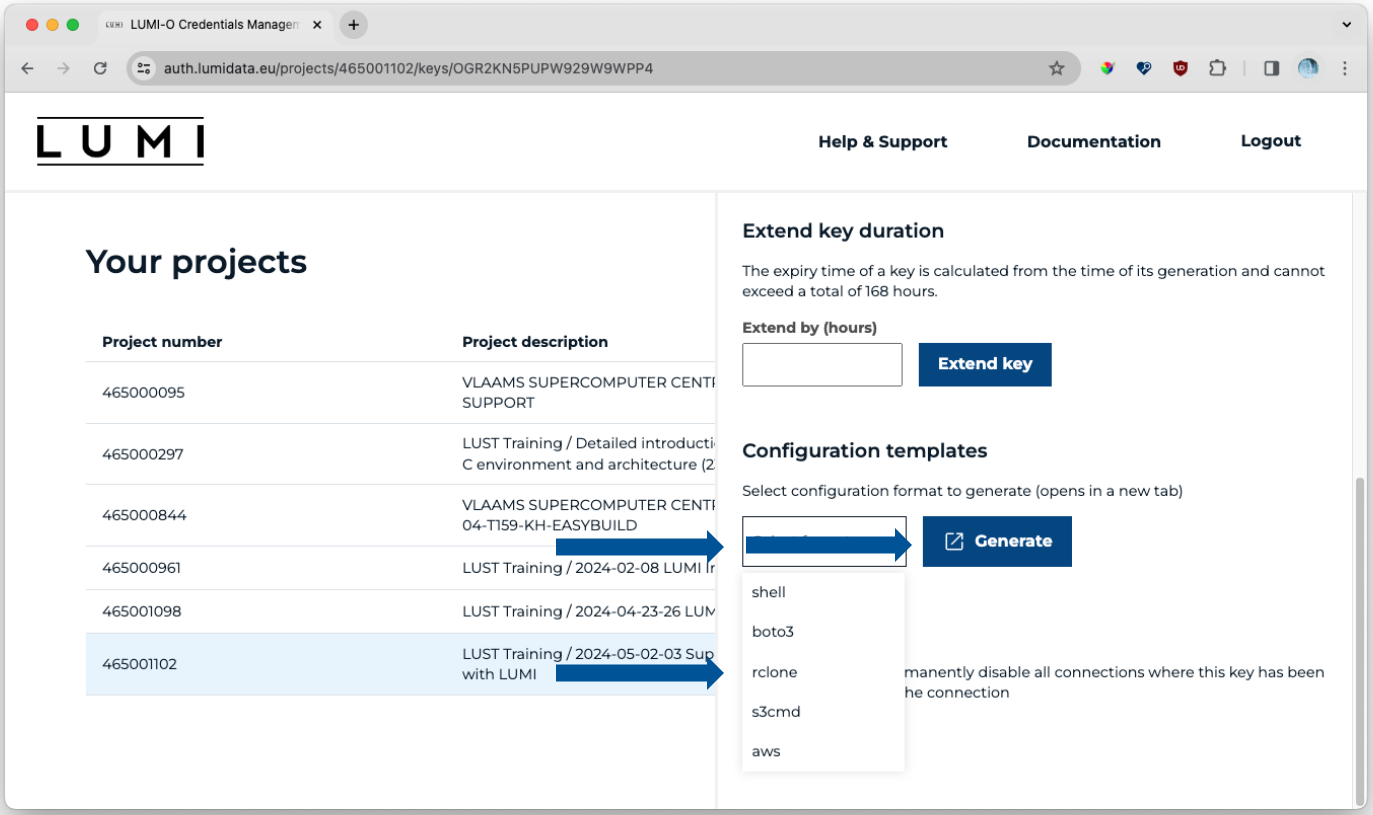
Credential mangement web interface: Check credentials



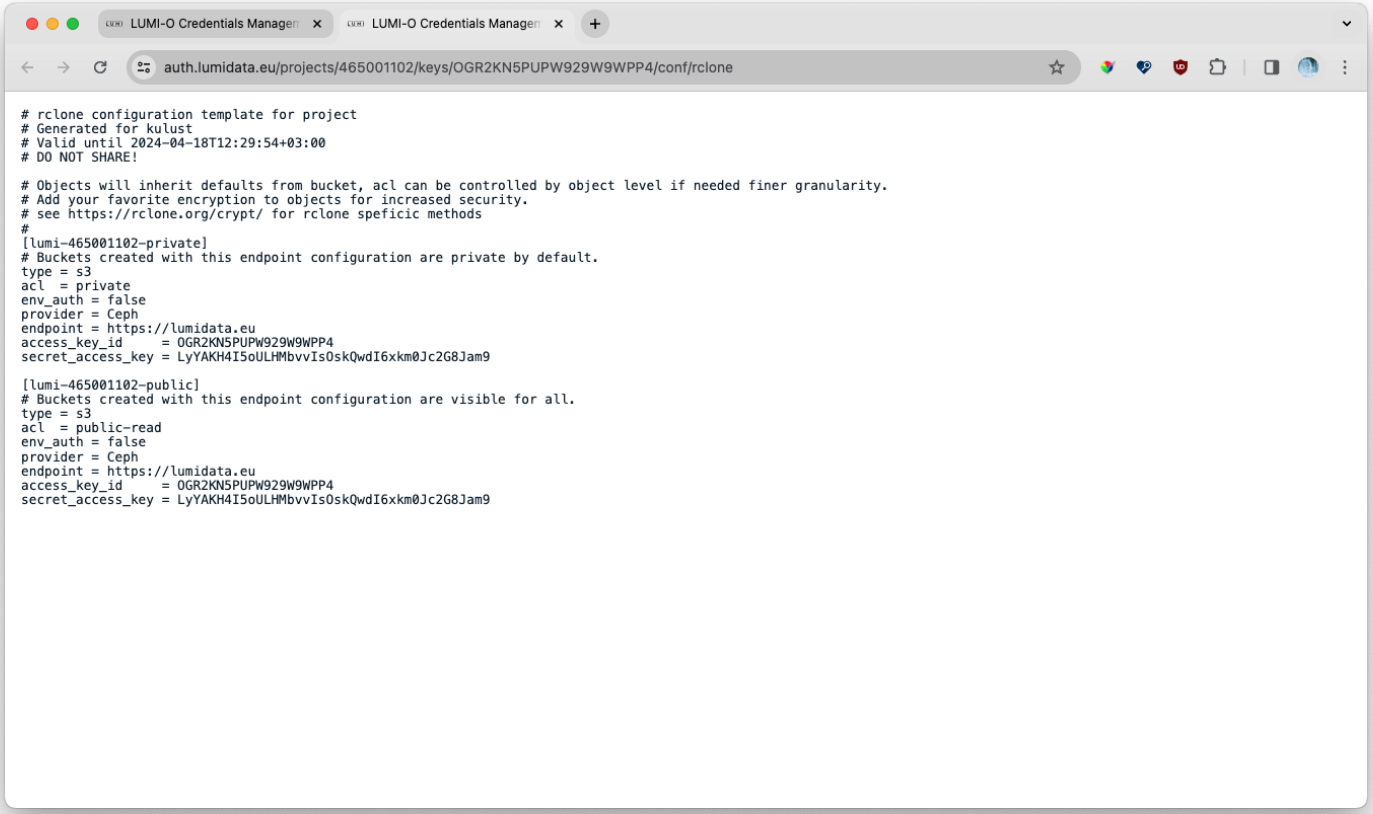
Credential mangement web interface: Extend credential lifetime



Credential mangement web interface: Tool configuration (1)



Credential mangement web interface: Tool configuration (2)



Credential management via OOD

CLI tool configuration on LUMI: lumio-conf

- On LUMI, you can use `lumio-conf` to configure `rclone` and `s3cmd`
  - Need to load the `lumio` module which also provides `rclone`, `s3cmd` and `restic`
  - Will ask for data from the "Access key details" screen
  - A future version may be more automatic
  - The `rclone` configuration differs from the one generated via the web interface
- Generate the configuration snippets via the web interface
  - E.g., for rclone: copy manually to `~/.config/rclone/rclone.conf`
  - Can be used to configure tools on your computer also

CLI tool configuration on LUMI: lumio-conf and rclone

- The `rclone` configuration contains two end points
  - With lumio-conf :
    - `lumi-o`: Buckets and objects uploaded to this endpoint will not be publicly accessible
    - `lumi-pub`: Buckets and objects uploaded to this endpoint will be publicly accessible
    - Can have both publicly accessible and not publicly accessible objects in one bucket
  - Web-generated configuration file:
    - `lumi-465001102-private`: Private buckets and objects
    - `lumi-465001102-public`: Public buckets and objects

Policies and ACLs

- Access control is managed through bucket policies and bucket and object access control lists (ACLs)
- Policies is a very powerful but also hard to use mechanism
  - Some information in the "Advanced usage of LUMI-O" section of the docs
  - And there is also information in the Ceph manual
  - Can be managed through `s3cmd`
- ACLs apply to individual buckets and objects
  - Can only add rights
  - Useful to make a bucket or object public, or give access to another project, but this is done to individual objects (unless applied recursively)

Policies and ACLs Examples

LUMI

- Make a bucket and all objects in it public or private
  ```
  s3cmd setacl --recursive --acl-public s3://bucket/
  s3cmd setacl --recursive --acl-private s3://bucket/
  ```
- Grant or revoke read rights to a bucket
  ```
  s3cmd setacl --acl-grant='read:465000000$465000000' s3://bucket
  s3cmd setacl --acl-revoke='read:465000000$465000000' s3://bucket
  ```
  - Note the use of single quotes to make sure that $465000000 is not interpreted as a variable name!
  - And similarly to objects
- Check the ACL and other information of a bucket or object
  ```
  s3cmd info s3://2day-20241210
  s3cmd info s3://2day-20241210/img/LUMI-2day-20241210-10-ObjectStorage/Title.png
  ```

Some tips & tricks

- When using the `rclone` command line tool, it is possible to throttle the speed for many commands with the `--bwlimit` command line option
  - May be needed if you upload from home over a very bandwidth-limited connection


LUMI
Questions?