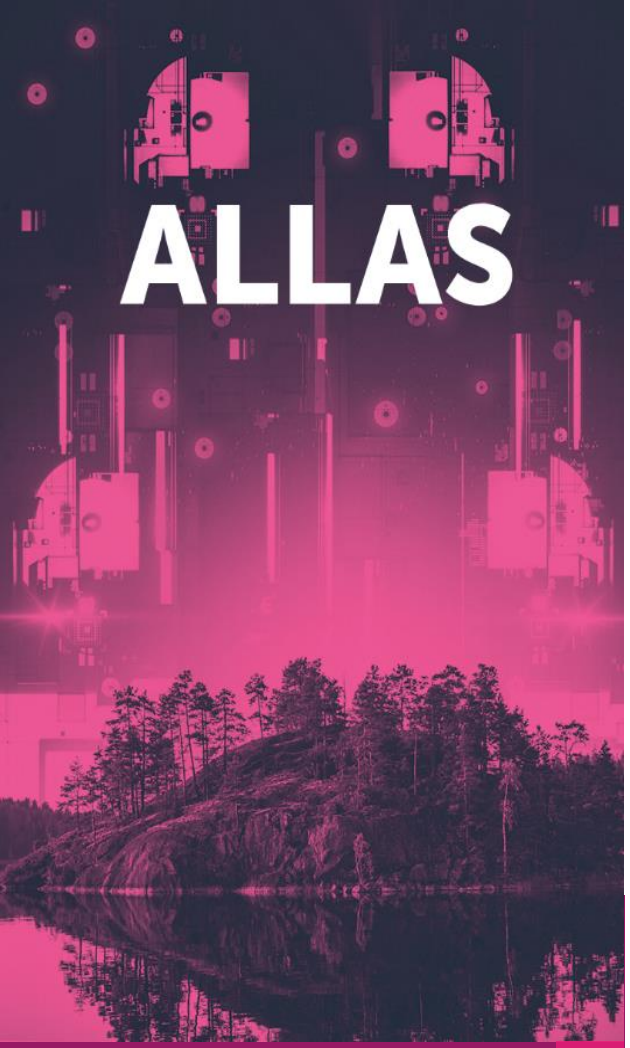
ALLAS AND GEOSPATIAL DATA

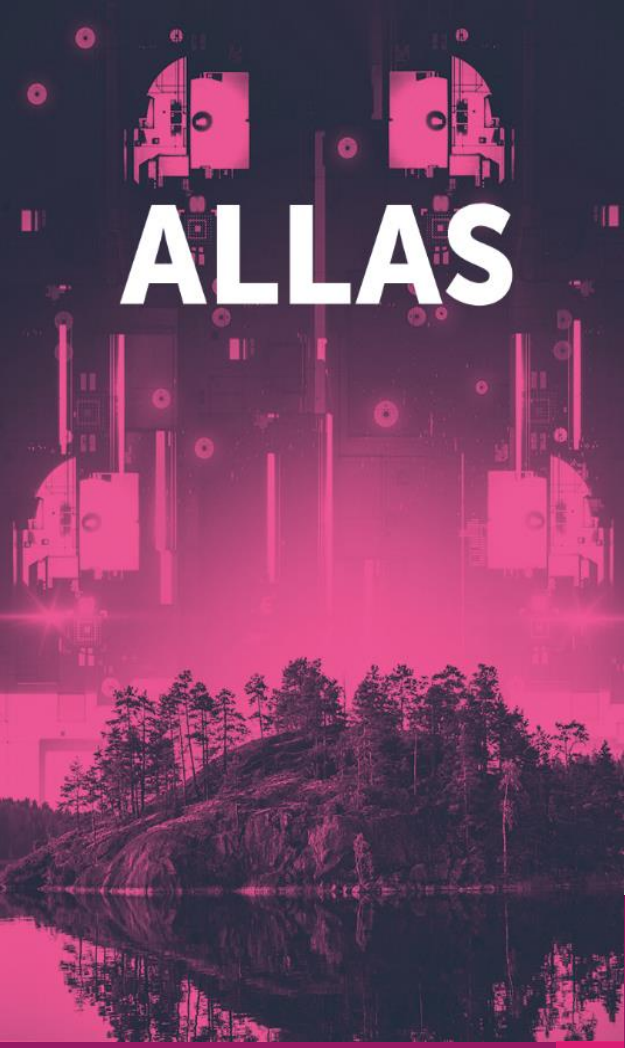
JOHANNES NYMAN, CSC

13.05.2020



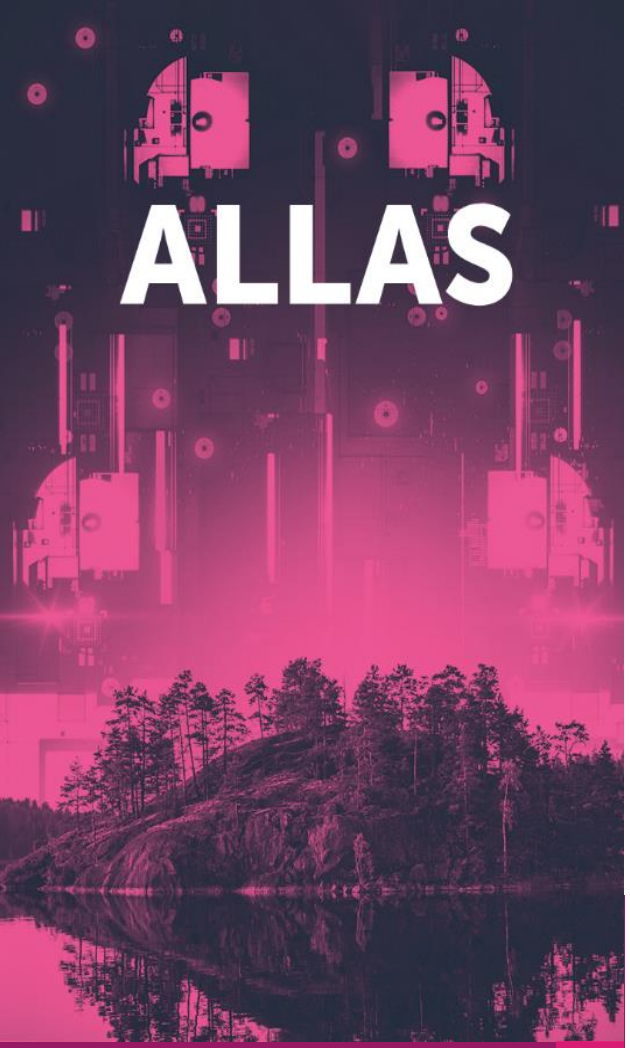
Welcome!

- This webinar will be recorded and published on the CSC YouTube channel
- If you have a question, you can write it in the chat window
- For a more comprehensive look into Allas, see previous webinars held by Kimmo Mattila
  - Data migration from Taito to Allas (1/2)



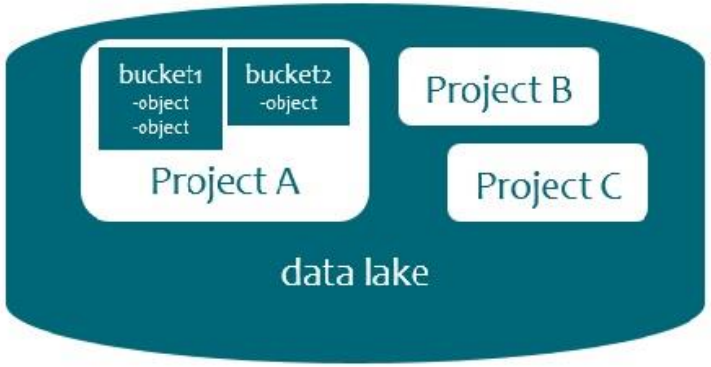
Allas – object storage: what it is for?

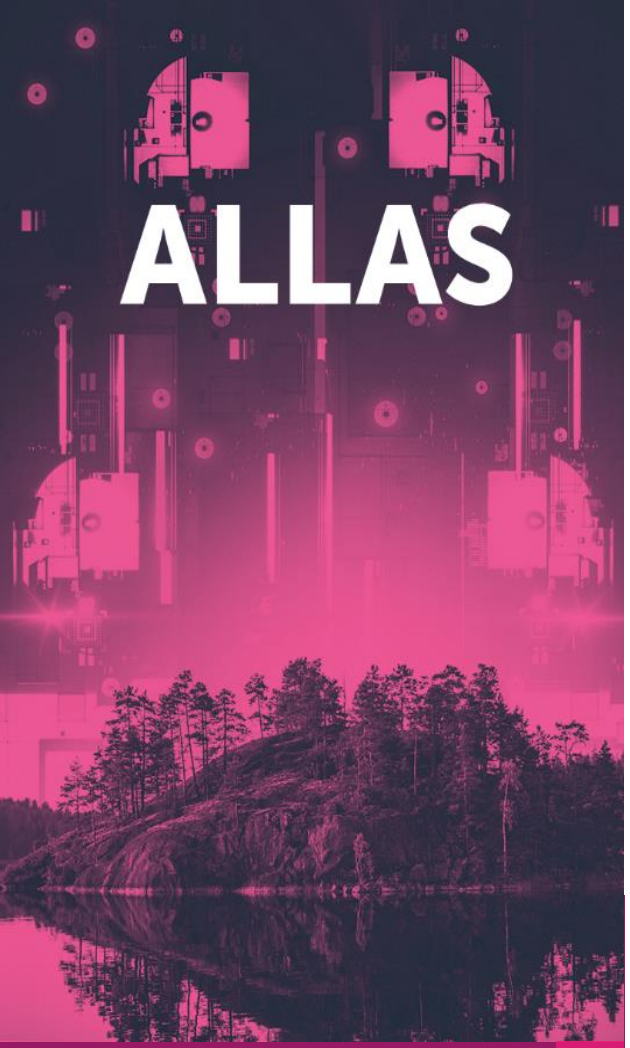
- Allas is a storage service for all computing and cloud services
- Meant for storing data during a project's lifetime
- Default quota 10 TB / project
- Possible to upload data from personal laptops or organizational storage systems
- Easily available in Puhti and Mahti
- Data can also be shared via Internet



Allas – object storage: terminology

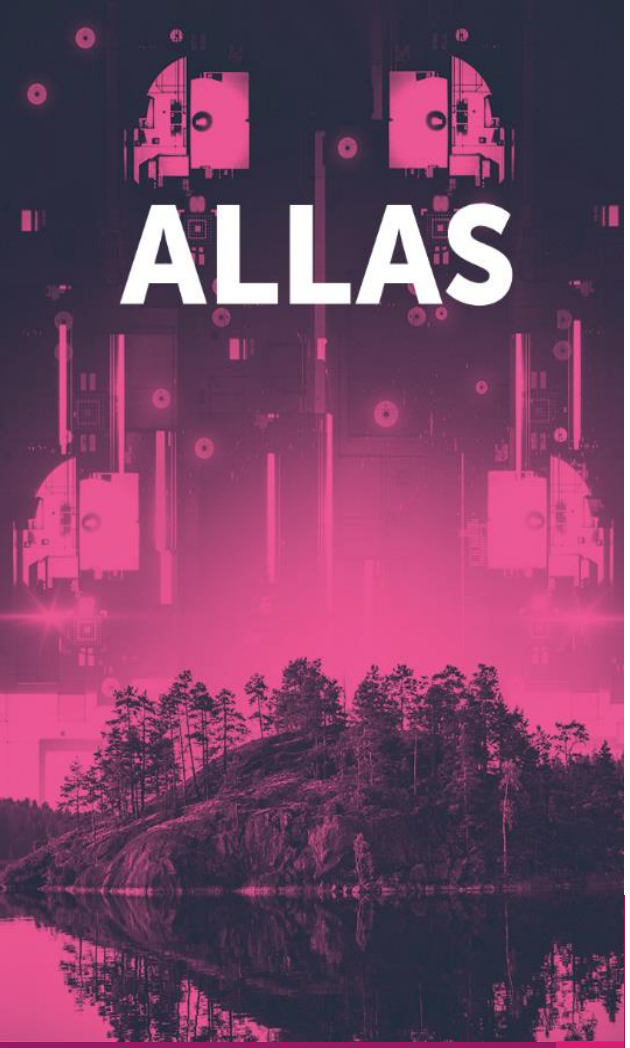
- Storage space in Allas is provided **per CSC project**
- Project space can have multiple **buckets** (up to 1000)
  - Only one level of hierarchy of buckets (no **buckets** within **buckets**)
  - Name of the **bucket** must be unique within Allas
- Data is stored as **objects** within a **bucket**
  - Blobs of data, can be anything (generally, **object** = file)
  - 500 000 objects / bucket
- Several levels of directory structure can be mimicked using **pseudofolders**





Allas – How to get access

- Register to CSC in https://my.csc.fi
- Set up a project in MyCSC
- Apply for Allas service, quota and billing units for your project
- Add other registered users to your project
- 10TB of storage space is now available for your project

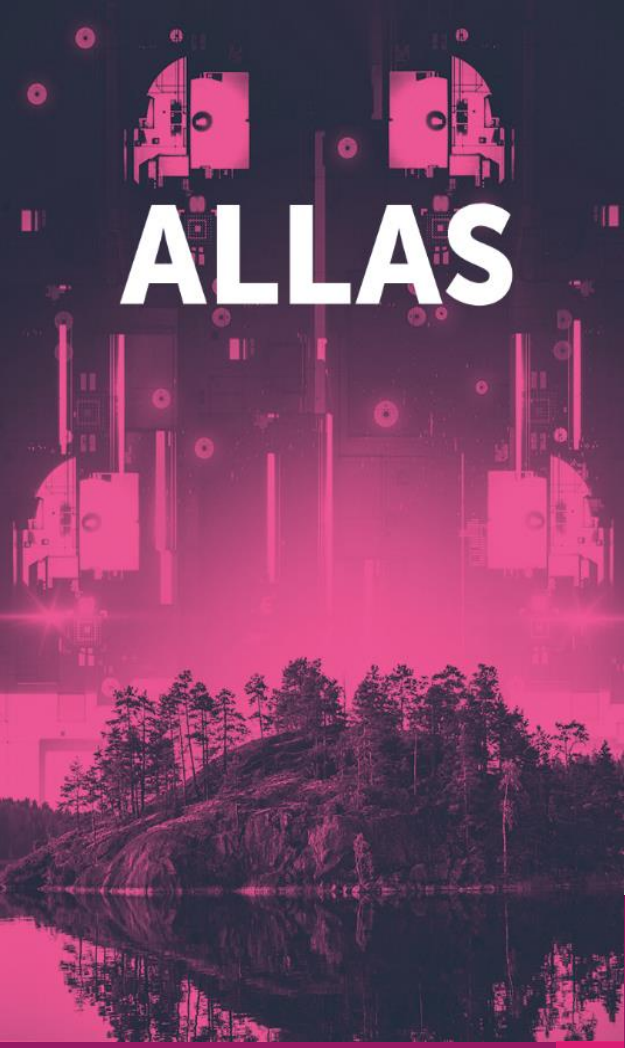


Allas supports Two Protocols

- **S3** (used by: s3cmd, rclone, WinSCP)
- **Swift** (used by: swift, rclone, a-tools, cyberduck)

- Authentication is different
  - **S3**: permanent key based authentication – nice, easy and less secure
  - **Swift:** authentication based on temporary tokens – more secure, requires authentication every 8 hours
- Metadata is handled in different ways
- Over 5G files are split in different ways

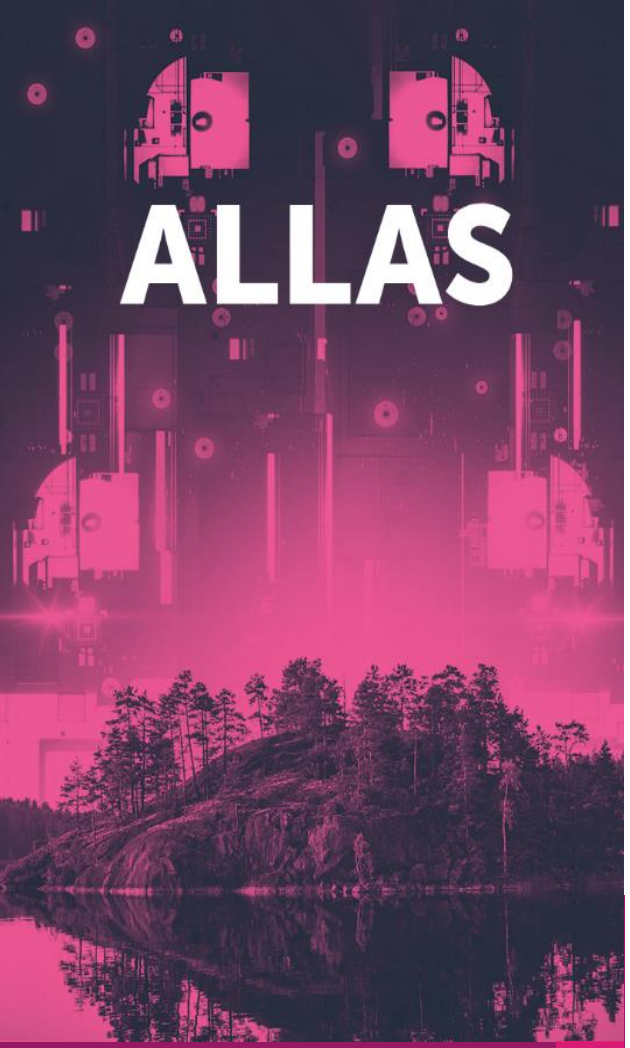
→ **We do not recommend cross-using Swift and S3 based objects!**



Allas – allas-conf

- The authentication command in Puhti
  - •Authenticates you with Allas using your CSC username/password
  - •Stores the token and auth keys in environmental variables from where the different tools can then access them
  - •Has two modes: one for **swift** (default) and one for **S3**
  - •Swift keys are valid for 8 hours, S3 keys do not expire and are stored in **.s3cfg** file
- If you want to use command line tools from somewhere else than Puhti, you need to populate the correct environmental variables/keys
  - •With **S3** permanent keys, you can use **allas-conf** in Puhti and copy the **.s3cfg** file to your local machine. After allas-conf, **.s3cfg** can be found from the home folder (hidden)
  - •The **allas-conf** command can also be installed locally https://github.com/CSCfi/allas-cli-utils

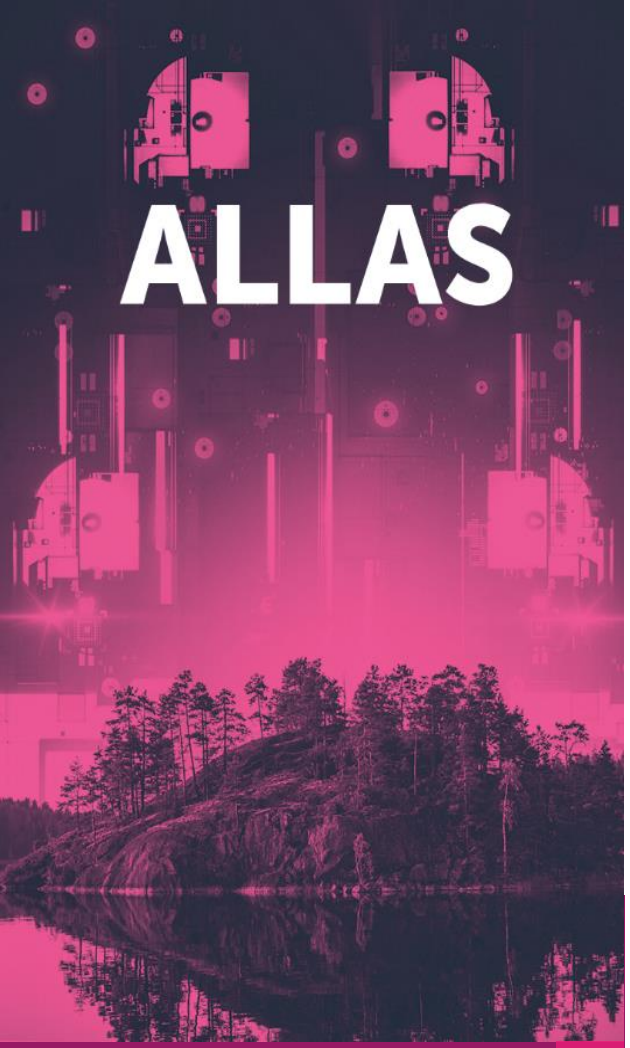
https://docs.csc.fi/data/Allas/accessing_allas/



Allas – rclone

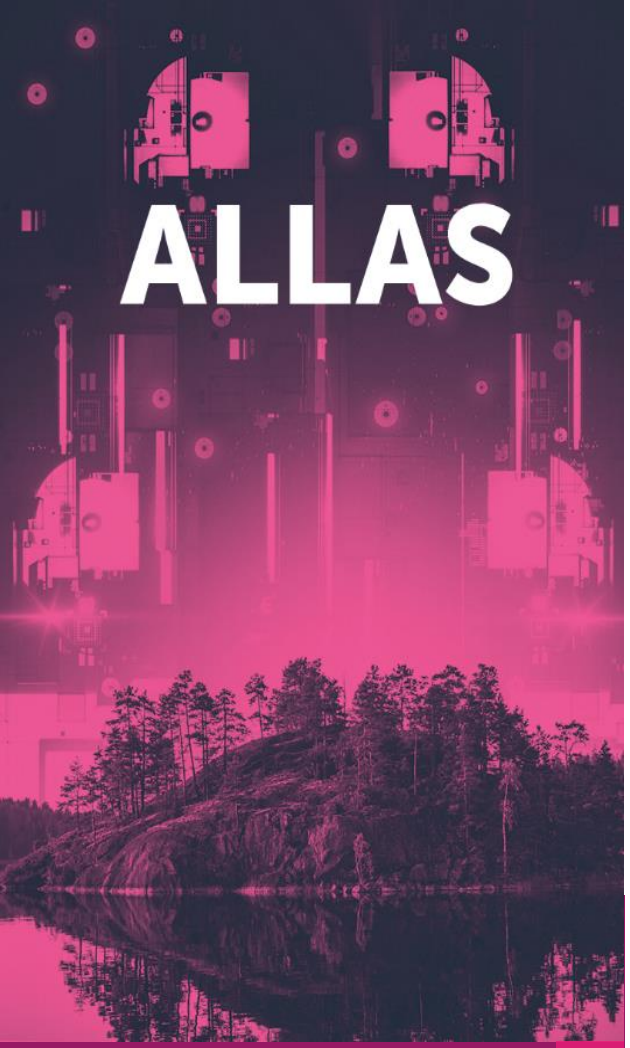
- Straight-forward power-user tool with wide range of features
- Fast and effective
- Available for Linux, Mac and Windows
- Overwrites and removes data without asking!
- The default configuration at CSC uses the swift-protocol but S3 can be used too

https://docs.csc.fi/data/Allas/using_allas/rclone



Graphical user interfaces for Allas

- **WinSCP -** Usually possible to install by your IT department
- **Cyberduck** – Free for Windows and Mac
- **Filezilla Pro** – Only Pro license supports object storages
- **Rclone built-in GUI –** Experimental. Not tested yet
- **Pouta.csc.fi -** Good for listing buckets and files and uploading single files
- **Windows File explorer mount with rclone**



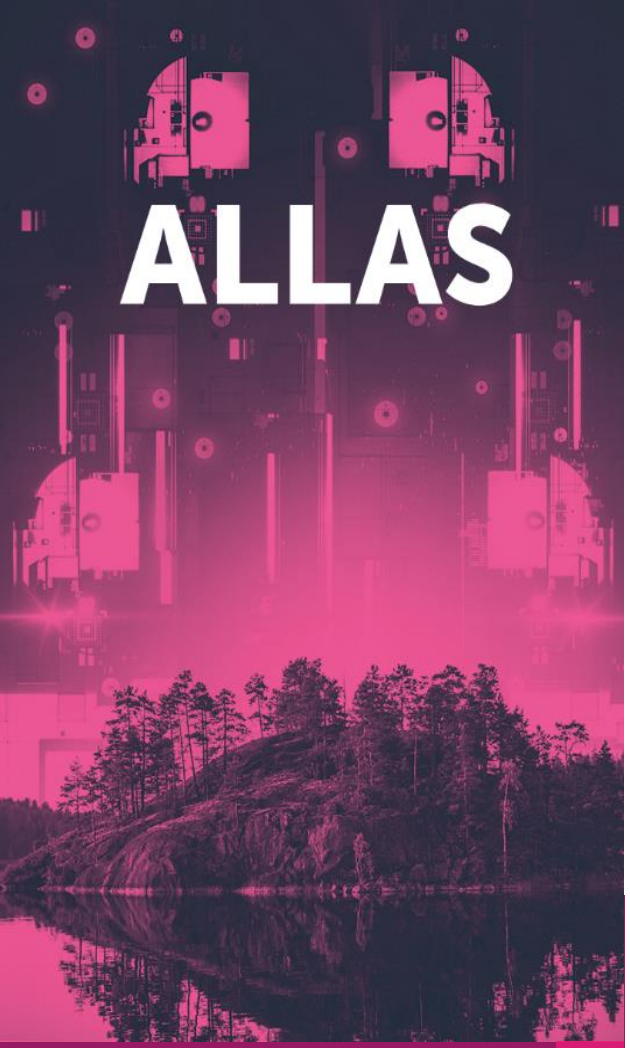
GDAL support for Allas

- GDAL support reading data from HTTP(S), S3 and SWIFT
- Tools based on GDAL support the same, inc R and Python
- GDAL does not support writing directly to object storage
  - Write locally to Puhti and then move to Allas with rclone/a-tools/etc
  - With R or Python also direct write is possible using additional libraries
- With big amount of not overlapping raster files using virtual rasters may be helpful.
  - Keep the raster files in Allas and only the .vrt in Puhti.
- https://docs.csc.fi/apps/gdal/

EXAMPLES & EXERCISES

JOHANNES NYMAN, CSC

13.05.2020



Reading GIS data directly from Allas with GDAL HTTPS

Load the Geoconda environment which includes GDAL

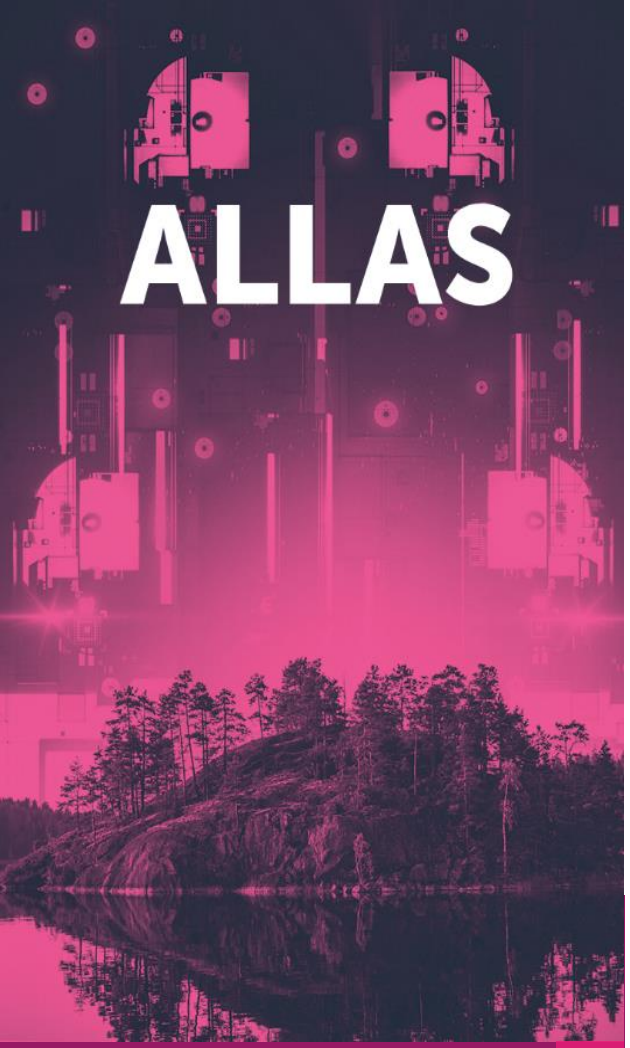
module load geoconda

Print info of a public image in Allas

gdalinfo /vsi**curl**/https://a3s.fi/gis-open/K3444D.tif

Or use any other GDAL command

gdal_translate /vsi**curl**/https://a3s.fi/gis-open/K3444D.tif local_K3444D.tif



Reading GIS data directly from Allas with GDAL S3

Setting up S3 connection

 # Needs to be run only once, or when changing project

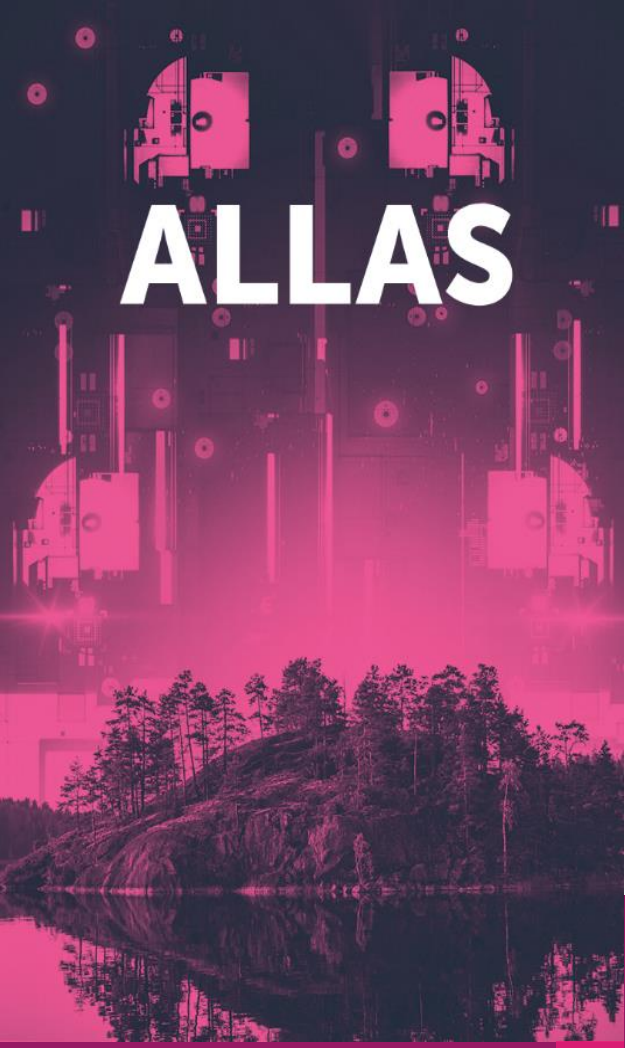
```
module load allas geoconda
allas-conf --mode s3cmd
```

Print info of a public image in Allas

```
gdalinfo /vsis3/gis-open/K3444D.tif
```

Or use any other GDAL command

```
gdal_translate /vsis3/gis-open/K3444D.tif  local_K3444D.tif
```



Reading GIS data directly from Allas with GDAL SWIFT

Setting up SWIFT connection

Needs to be run every 8 hours

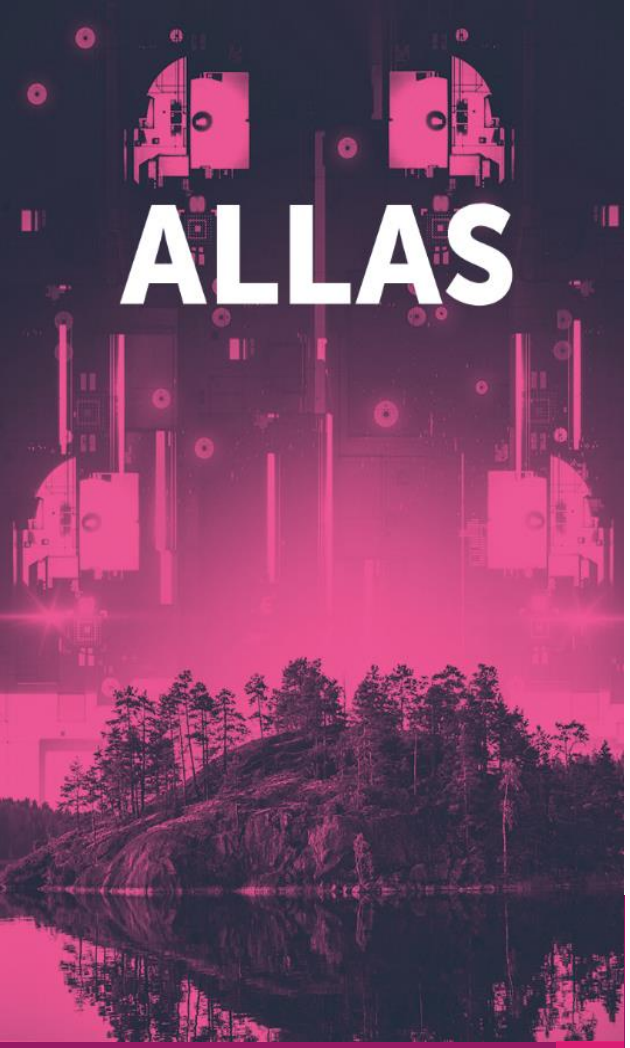
```
module load allas geoconda
allas-conf
export SWIFT_AUTH_TOKEN=$OS_AUTH_TOKEN
export SWIFT_STORAGE_URL=$OS_STORAGE_URL
```

Print info of a public image in Allas

```
gdalinfo /vsi**swift**/gis-open/K3444D.tif
```

Or use any other GDAL command

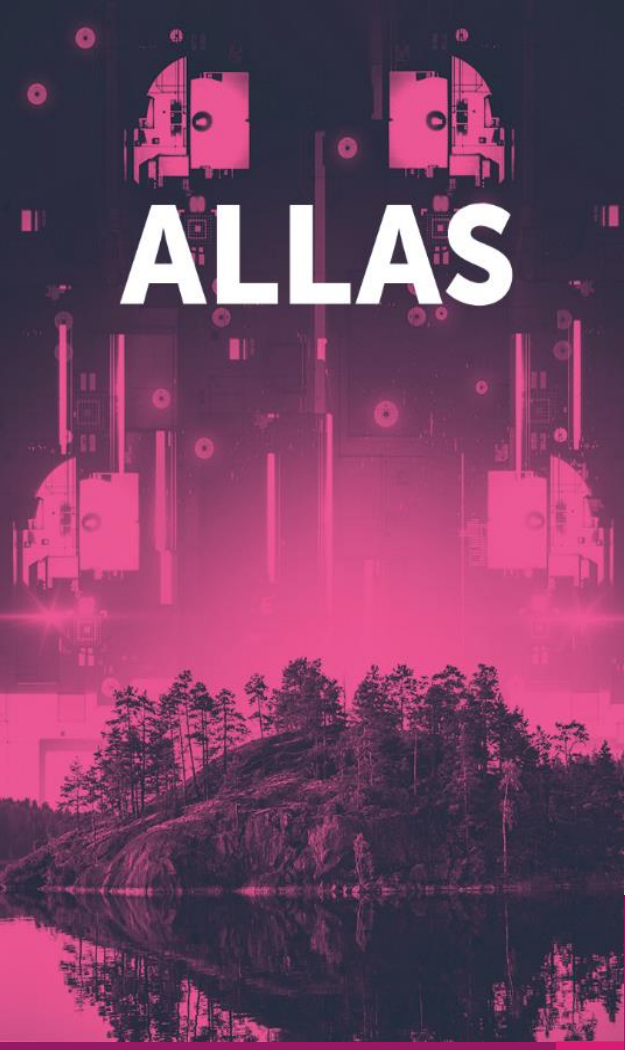
```
gdal_translate /vsi**swift**/gis-open/K3444D.tif  local_K3444D.tif
```



Example dataset

You can use the same example dataset if logged in Puhti
Just create the necessary folders and copy the files

```
mkdir -p ~/webinar/results

cp /appl/data/geo/mml/dem2m/2008_latest/L2/L24/*.tif ~/webinar/results

cd ~/webinar
```



Python and S3 1/3

Start by running allas-conf in S3 mode (Swift mode is default)

```
module load allas
allas-conf --mode s3cmd
```

Load the geoconda module and start Python on the login node

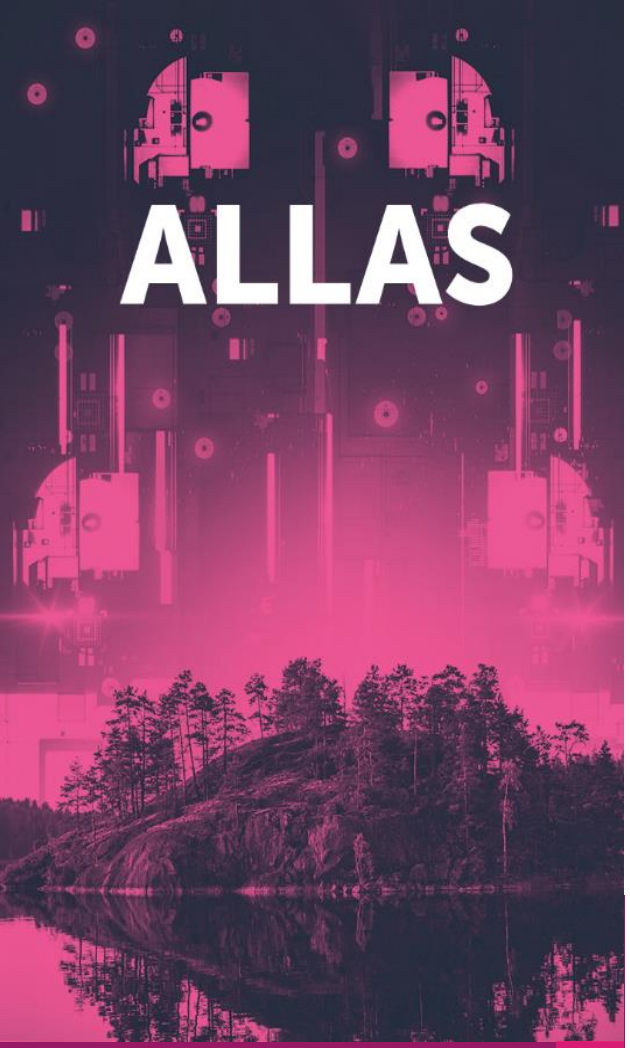
```
module load geoconda
python
```

Import necessary libraries

```
import os
import boto3
```

The filepath to the folder where the files are and the name of our bucket

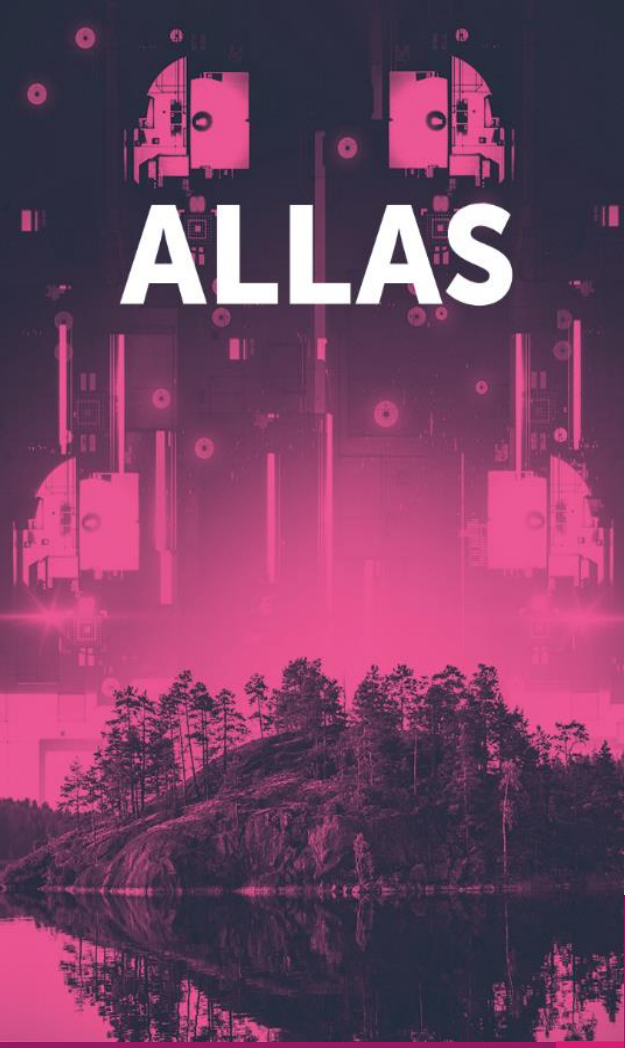
```
results_folder = '/users/johannes/webinar/results'
our_bucket = 'webinar'
```



Python and S3 2/3

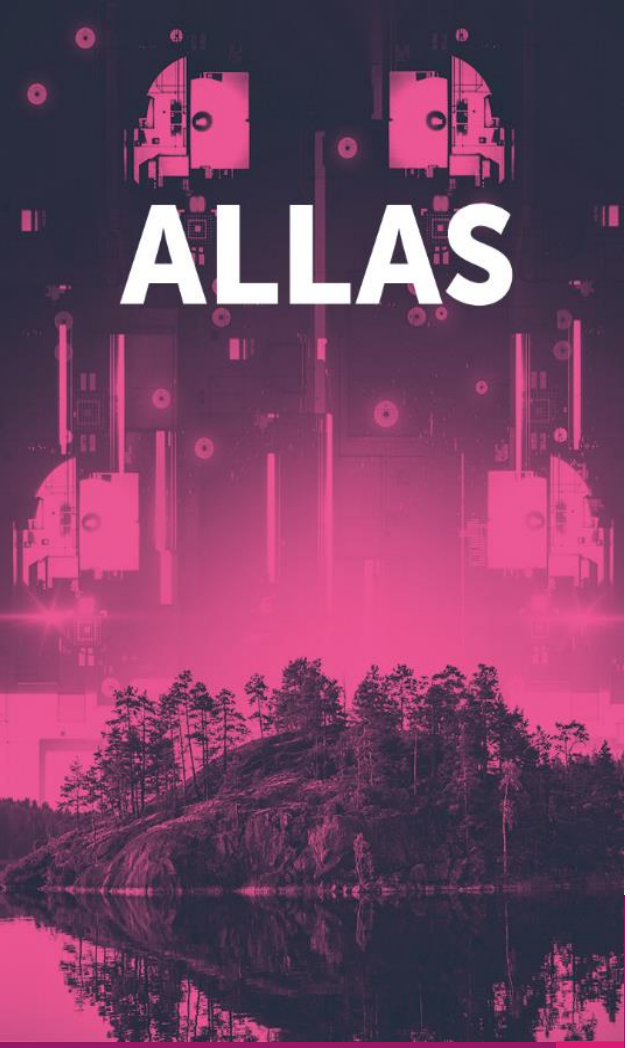
```
## Establish S3 connection to Allas

s3 = boto3.client('s3',  endpoint_url='https://a3s.fi')
s3.create_bucket(Bucket=our_bucket)

## Uploading files to Allas (and a creating bucket in the process)

for filename in os.listdir(results_folder):
    full_filepath = os.path.join(results_folder, filename)
    s3.upload_file(full_filepath, our_bucket , 'results/' + filename)

## Now you can loop the files in the bucket and print their filenames

for object in s3.list_objects(Bucket=our_bucket )['Contents']:
    print(object ['Key'])

## Download the files back to Puhti overwriting the current ones

for object in s3.list_objects(Bucket=our_bucket)['Contents']:
    s3.download_file(our_bucket , object['Key'], object['Key'])
```

Exit Python with quit()



Python, rasterio and S3 3/3

```
## Saving a raster to Allas with rasterio

import rasterio
from rasterio.io import MemoryFile

## Read a raster to a rasterio object


r = rasterio.open (results_folder+ '/L2433A.tif')
input_data = r.read()

## Create the raster file to memory and write to Allas

with MemoryFile() as mem_file:
   with mem_file.open(**r.profile) as dataset:
      dataset.write(input_data)
   s3.upload_fileobj(mem_file, our_bucket, 'results/L2433E_fromPython.tif')
```



R and S3 - Preparations 1/4

Unload previous modules and load the R module

```
module unload geoconda
module load allas r-env-singularity
```

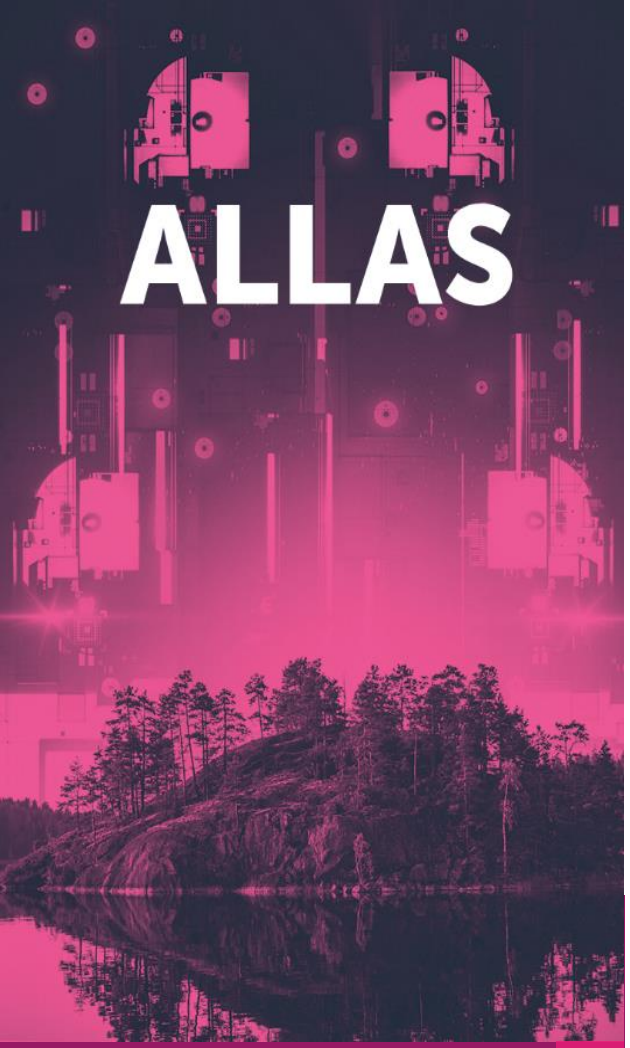
Run allas-conf and start r session on login node

```
allas-conf --mode s3cmd
singularity_wrapper exec R --no-save
```

Load the aws.s3 library and set the folder filepath

```
library("aws.s3")
```

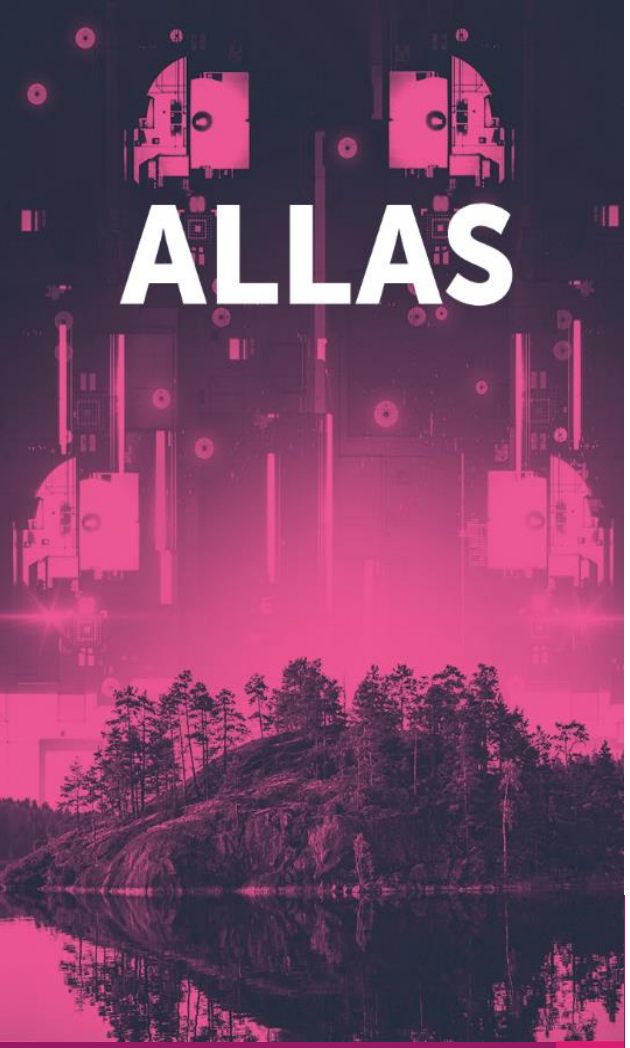
```
webinar_folder <- "/users/johannes/webinar"
results_folder <- "/users/johannes/webinar/results"
```



R and S3 – Syncing folders 2/4

```
## r-env-singularity has newer aws.s3 library, with better s3sync function

## List all files in our bucket

our_bucket <- "webinar"
bucket <- get_bucket(our_bucket, region="")
bucket

## Download them one by one to the results folder

s3sync(webinar_folder, our_bucket, direction = "download", region="")

## Remove them from Allas

for ( object in bucket ){  delete_object(object, region="")  }

## Upload them back to Allas

s3sync(webinar_folder, our_bucket, direction = "upload", region="")
```

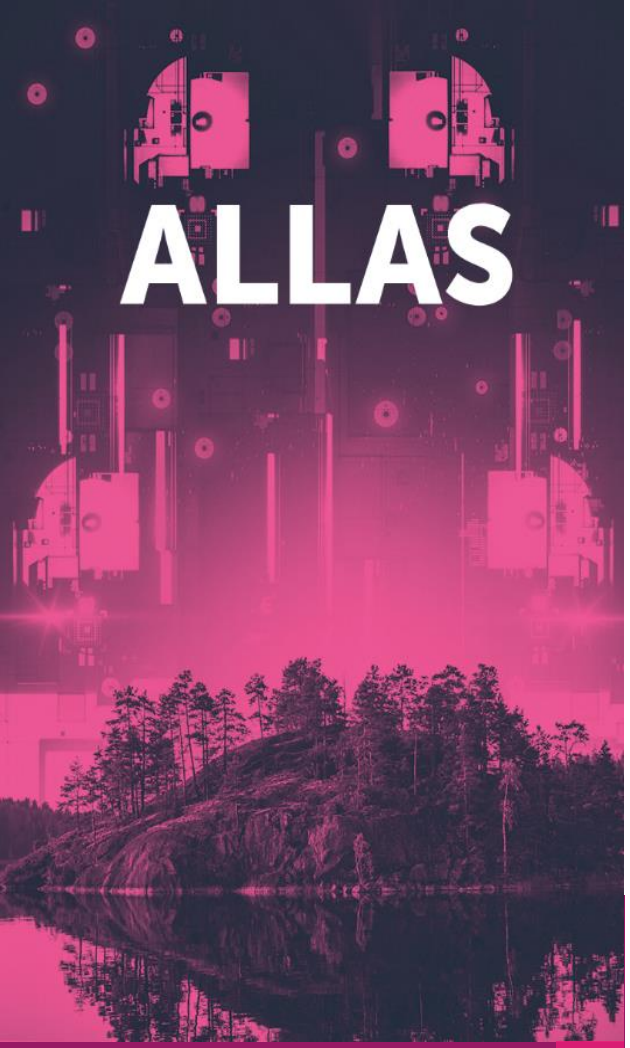
Exit R with q()



R and S3 – Download/Upload single files 3/4

```
## r-env has older aws.s3 library which requires looping files

## Download them one by one to the results folder

bucket <- get_bucket(our_bucket, region="")
for ( object in bucket ){  save_object(object, file=object["Key"][[1]], region="")  }


## Upload them back to Allas

files <- list.files( results_folder, full.names=TRUE )
for (file in files) {
   put_object(file, file.path("results", basename(file)), bucket, region="")  }
```

Exit R with q()



R and S3 – Writing from memory to Allas 4/4

```
## Load raster library

library("raster")

## Read the raster

r <- raster(file.path(results_folder, "L2433A.tif"))

## Write the raster straight from memory to Allas


s3write_using(r, FUN = raster::writeRaster, bucket = our_bucket,
object="results/L2433A_fromR.tif", opts=c( region = "" ))
```

WinSCP

For WinSCP connection you need the S3 access key and secret key. If you don't have them yet somewhere, you can run this again or find them from the .s3cfg file

allas-conf --mode s3cmd

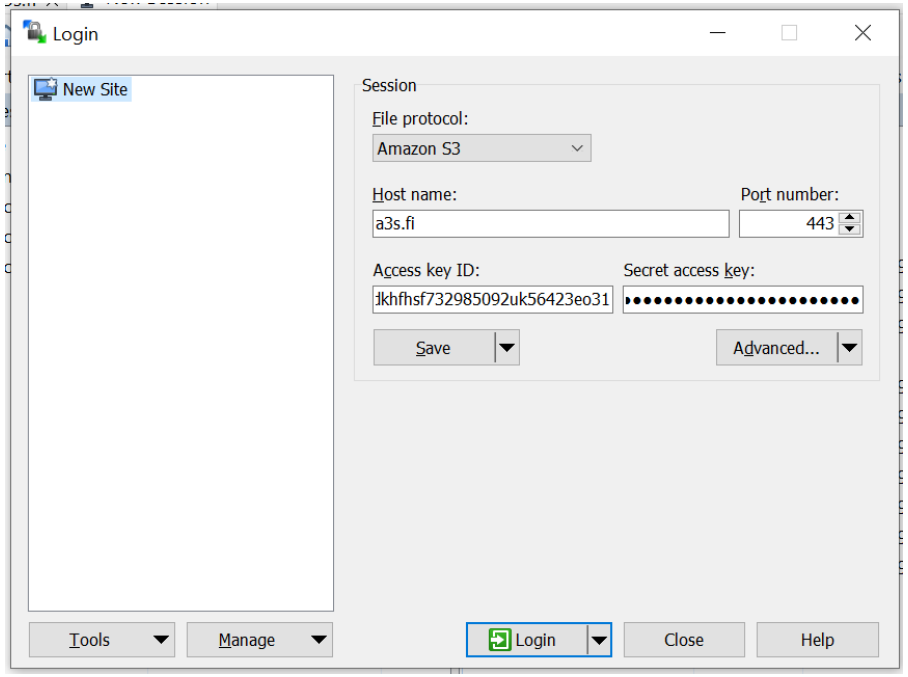
After choosing the project it prints the access key and secret key. Copy them and paste to WinSCP. Change also the other input fields accordingly

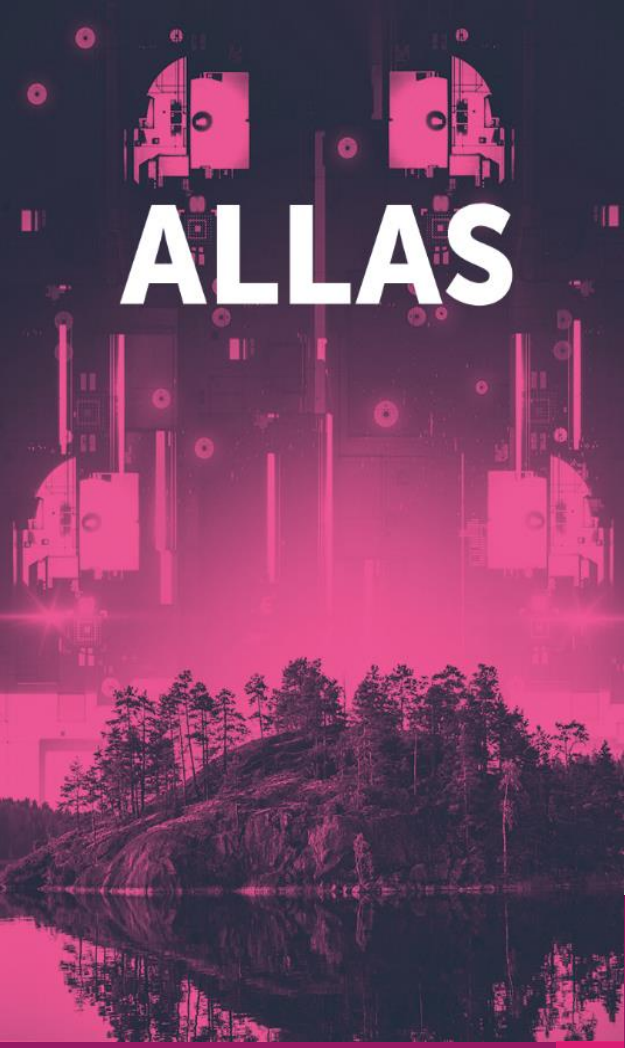
Host name: **a3s.fi**

Port number: **443**

Access Key: <your key>

Secret access Key: <your key>





Reading GIS data directly from Allas with a VRT

- In cases where you have very large quantities of raster data in Allas, it is beneficial to access it through a GDAL virtual raster
- You can create a file list of all .tif files inside a bucket with

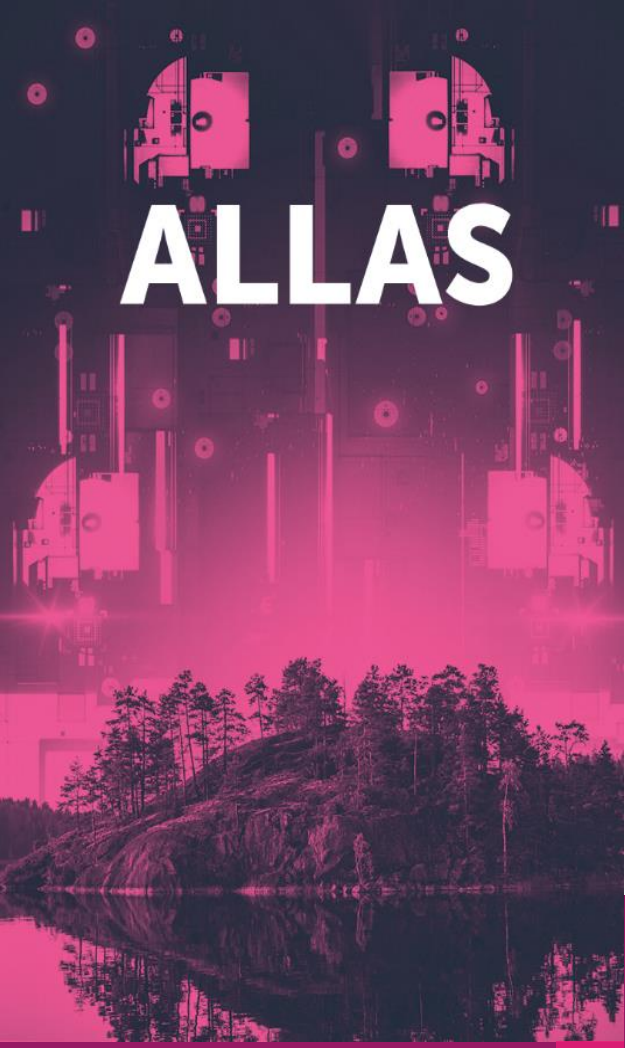
```
allas-conf
rclone lsf --include '*.tif' allas:webinar/results > file_list.txt
sed -i -e 's-^-/vsis3/webinar/results/-' file_list.txt
```

- Then create the virtual raster file

```
gdalbuildvrt -input_file_list file_list.txt dems.vrt
```

- Now you can clip a bbox from the vrt file without worrying which files it intersects

```
gdal_translate -projwin 106600 6715000 107100 6714800 dems.vrt clipped_dem.tif
```



Useful links

- More on GDAL:

https://research.csc.fi/gdal_ogr

- More on Virtual Rasters:

https://research.csc.fi/virtual_rasters

- Our Python code examples

https://github.com/csc-training/geocomputing/tree/master/python

- Our R code examples

https://github.com/csc-training/geocomputing/tree/master/R

- General CSC documentation (Puhti, Allas, Pouta, Rahti)

https://docs.csc.fi/

THANK YOU!

JOHANNES NYMAN, CSC

13.05.2020