VSC HPC Introduction

ICTS KU Leuven
VSC staff:
Ehsan Moravveji
Mag Selwa
Wouter van Assche
Geert Jan Bex (Uhasselt)
Jan Ooghe

Material

- Everything is on Github: https://hpcleuven.github.io/HPC-intro/
- Video Recordings
  - Scan the QR code
  - Recommended videos: ~ 2 hrs
  - Optional videos: ~1 hr

What is High Performance Computing?

- Using supercomputers to solve advanced computation problems
- Reduce the computation time from days, years, decades, or centuries to minutes, hours, days, or weeks
- The key is parallelism
- Access to specialized hardware (GPU, large memory, high-speed interconnect)

In practice, it is more like …

The concept is simple: **Parallelism** = employing multiple processors for a single problem

Outline

Part 1: Basics

- ❑ VSC
- ❑ Tier-2 Clusters
- ❑ Storage
- ❑ Accounting
- ❑ Login
  - Open OnDemand
  - SSH client & Terminal
  - NX
- ❑ Data Transfer
  - FileZilla
  - rsync
  - Globus

Part 2: Hands-on

- ❑ Using Open OnDemand
- ❑ Software
- ❑ Miniconda
- ❑ Batch Jobs
- ❑ Worker
- ❑ Demo: test yourself

VSC

(Vlaams Supercomputer Centrum)

VSC PARTNERSHIP

Supported by

fwo

ASSOCIATIE UNIVERSITEIT GENT

universitaire associatie BRUSSEL

ASSOCIATIE KU LEUVEN

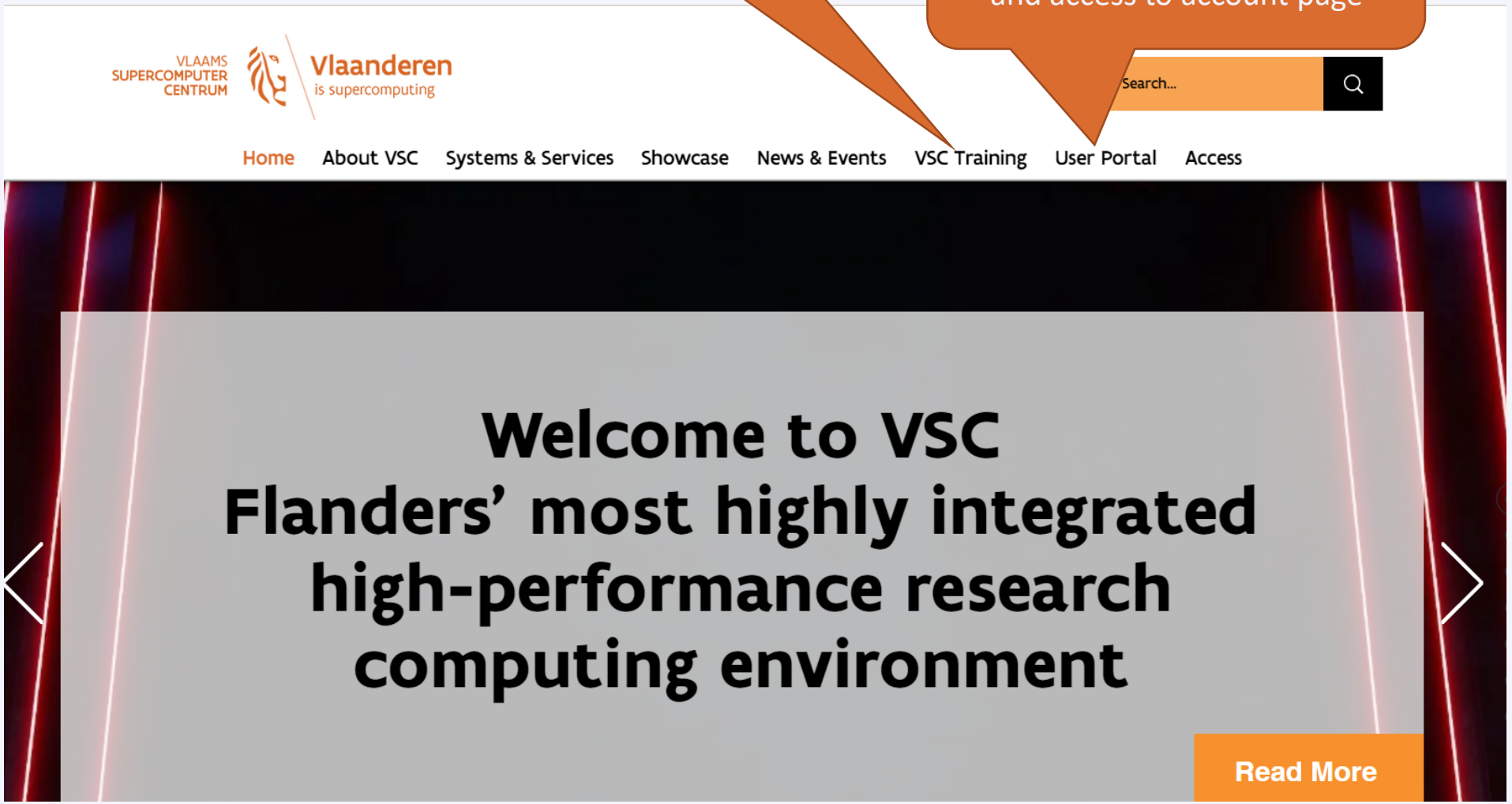
ASSOCIATIE UNIVERSITEIT-HOGESCHOLEN LIMBURG

auha antwerp university association

Surf to: www.vscentrum.be

Very rich suite of courses every academic year

Documentation!
Answers >80% of your questions and access to account page



VLAAMS SUPERCOMPUTER CENTRUM

Vlaanderen
is supercomputing

Search...

Home | About VSC | Systems & Services | Showcase | News & Events | VSC Training | User Portal | Access

Welcome to the User Portal

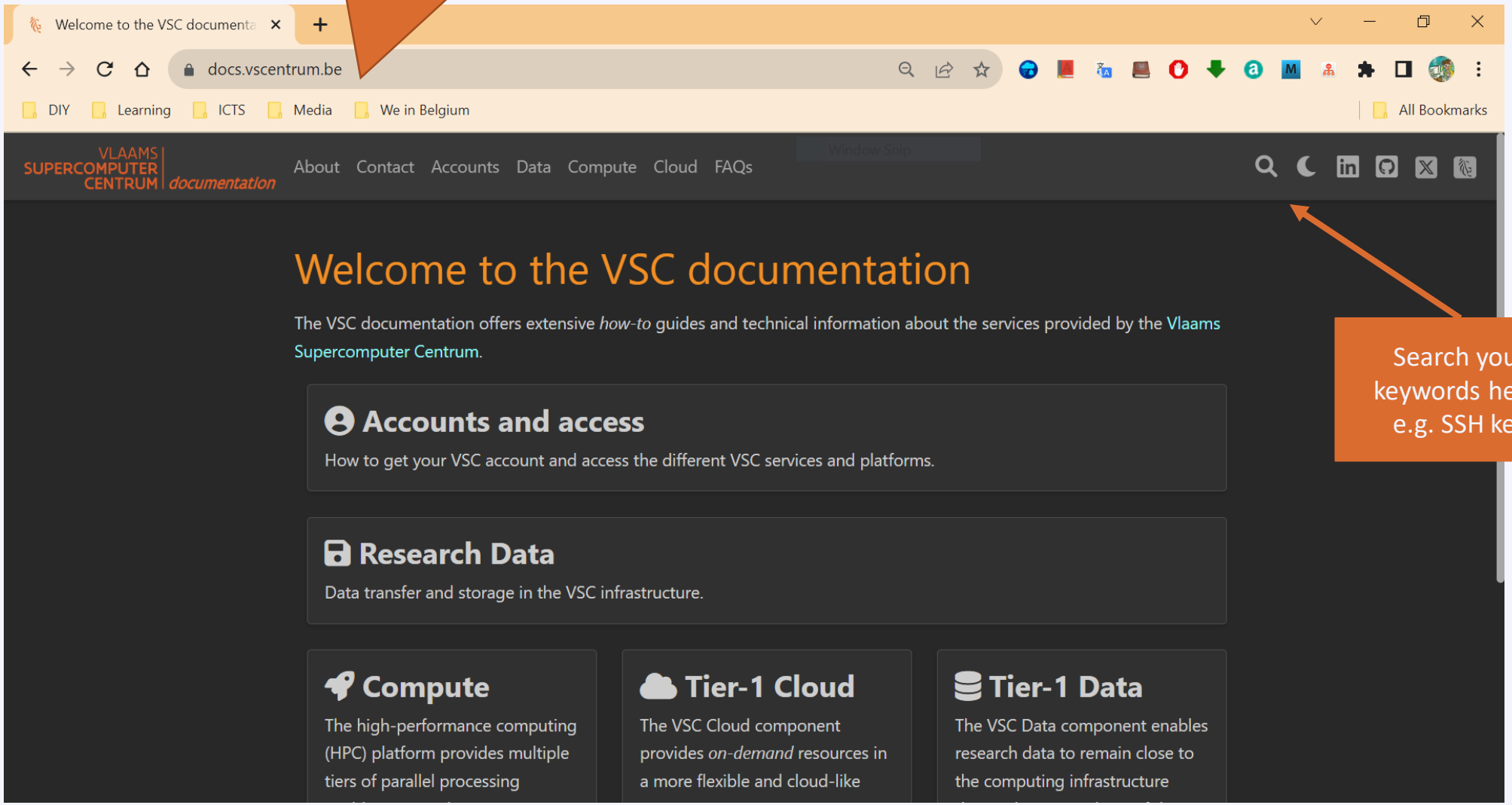
Here you can find the gateway to the User documentation of the Vlaams Supercomputer Centrum

User documentation page

Manage your VSC-account
partner institute account required

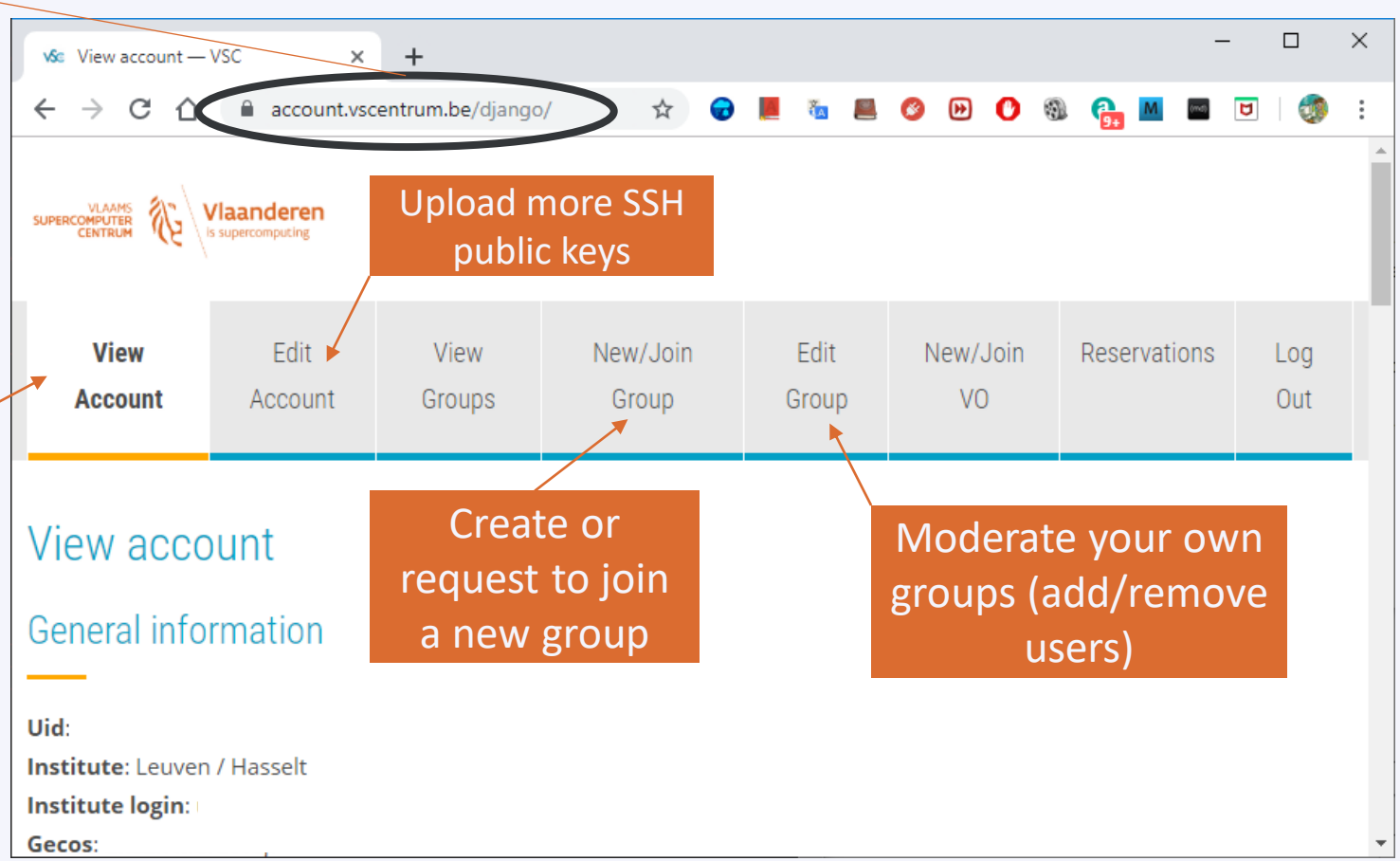
VSC account page

https://docs.vscentrum.be



Search your keywords here; e.g. SSH key

To manage your VSC account: account.vscentrum.be



Upload SSH Key
Request account

Create or request to join a new group

Moderate your own groups (add/remove users)

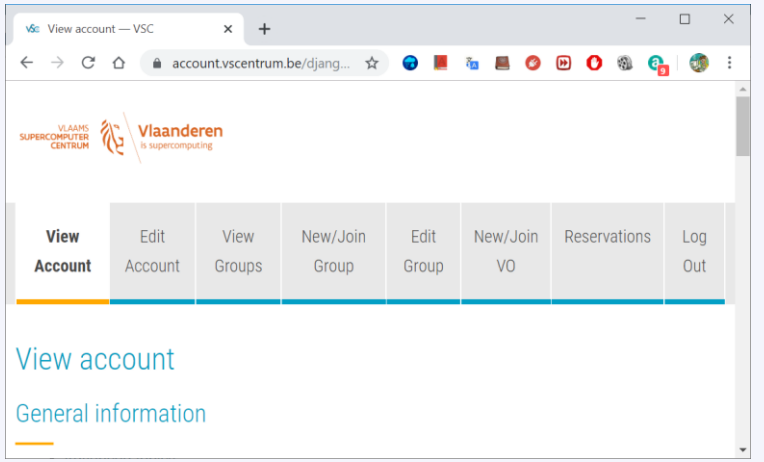
VLAAMS SUPERCOMPUTER CENTRUM

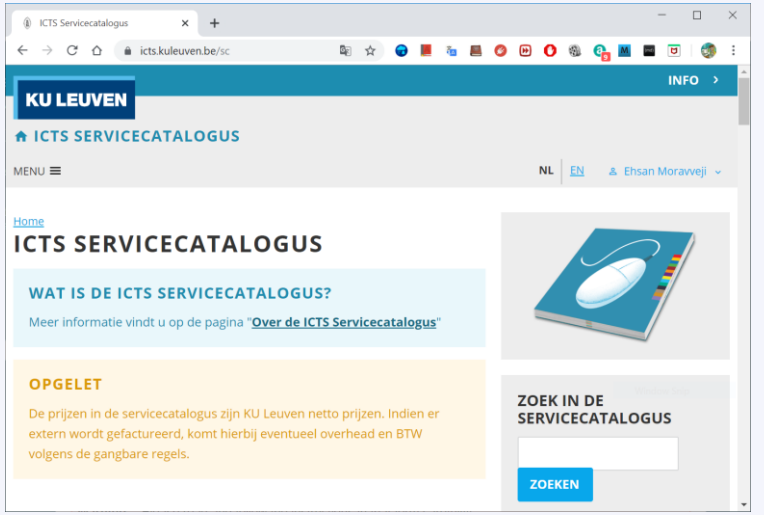
Support and Services

Basic support	• Helpdesk (hpcinfo@kuleuven.be) • Monitoring and reporting
Application support	• Installation and porting • Optimisation and debugging • Benchmarking • Workflows and best practices
Training	• Documentation and tutorials • Scheduled trainings / workshops • On request workshops • One-to-one sessions

Become a VSC user

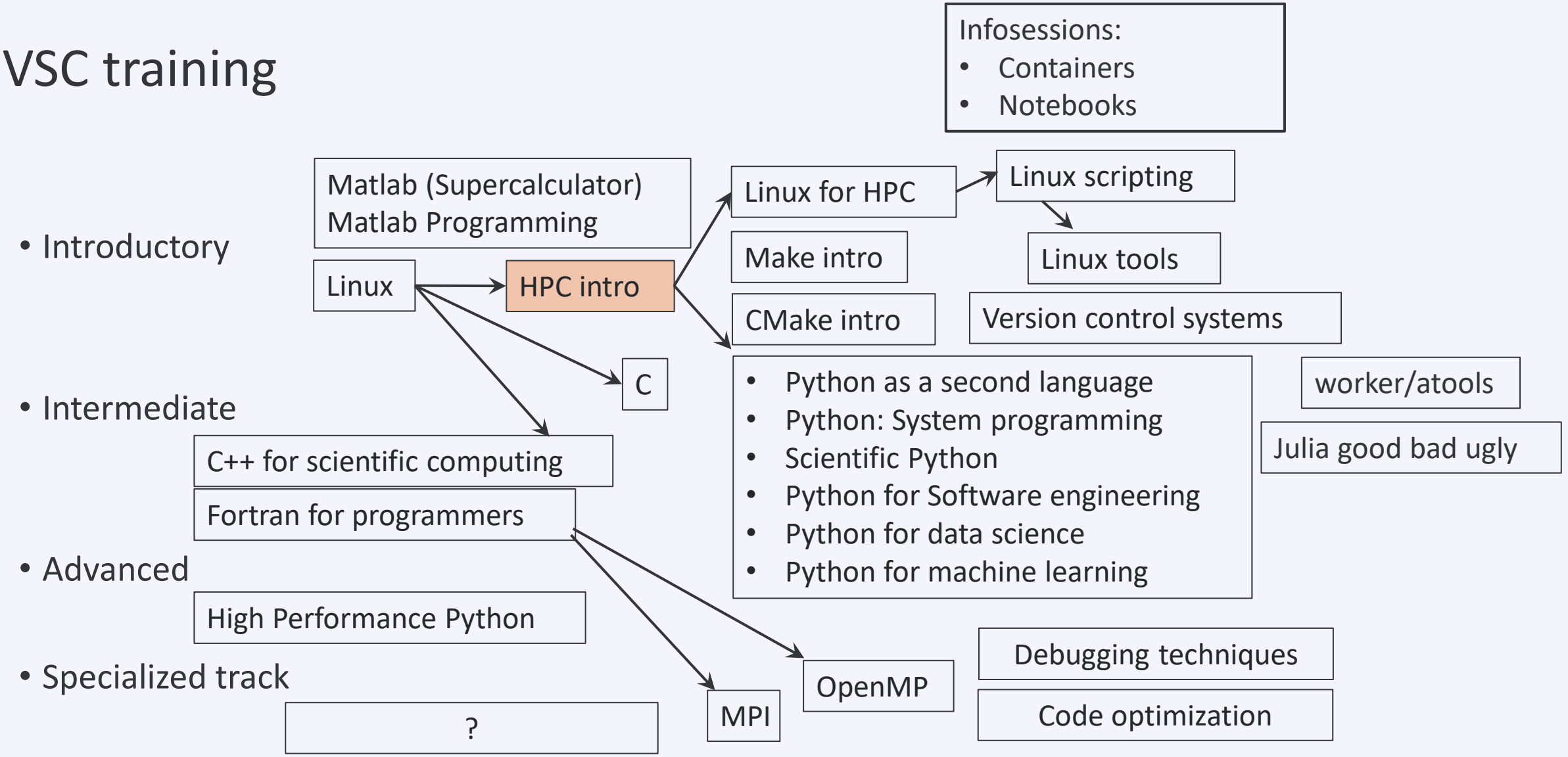
- (Optional) Create a secure (4096 bit) SSH key pairs
  Upload it on the account page: www.account.vscentrum.be
- You need to request a VSC account
  Normally processed swiftly
- Request introductory credits (2M free credits for 6 months)
- Request project credits (for supervisors and project leaders)
  You need to create a VSC group
  Add users to the group to give them access to use credits
  Fill out the request form
- Extra storage requests
  Scratch extension: free of charge
  Staging fileset: 20 € per TB per year
- All service costs (compute and storage) are all explained
  Go to ICTS service catalogus: https://icts.kuleuven.be/sc
  Click on High Performance Computing (NL/EN)





VLAAMS SUPERCOMPUTER CENTRUM

VSC training



Stay up-to-date https://www.vscentrum.be/en/education-and-trainings

To Acknowledge VSC in publications

Why?

- ❑ a contractual obligation for the VSC
- ❑ helps VSC secure funding
- ❑ you will benefit from it in the long run

At KU Leuven

- ❑ add the relevant papers to the virtual collection "High Performance Computing" in Lirias

In het nederlands

De rekeninfrastructuur en dienstverlening gebruikt in dit werk, werd voorzien door het VSC (Vlaams Supercomputer Centrum), gefinancierd door het FWO en de Vlaamse regering – departement EWI.

In English

The computational resources and services used in this work were provided by the VSC (Flemish Supercomputer Center), funded by the Research Foundation - Flanders (FWO) and the Flemish Government – department EWI.

Tier-2 Clusters

VSC HPC Environments

Tier-0

PRACE

Vlaanderen
is supercomputing

Tier-1

Hortense

Tier-2

GHENT UNIVERSITY — **STEVIN**

VUB VRIJE UNIVERSITEIT BRUSSEL — **HYDRA**

Universiteit Antwerpen — **HOPPER/LEIBNIZ**

KU LEUVEN UHASSELT — **GENIUS/WICE**

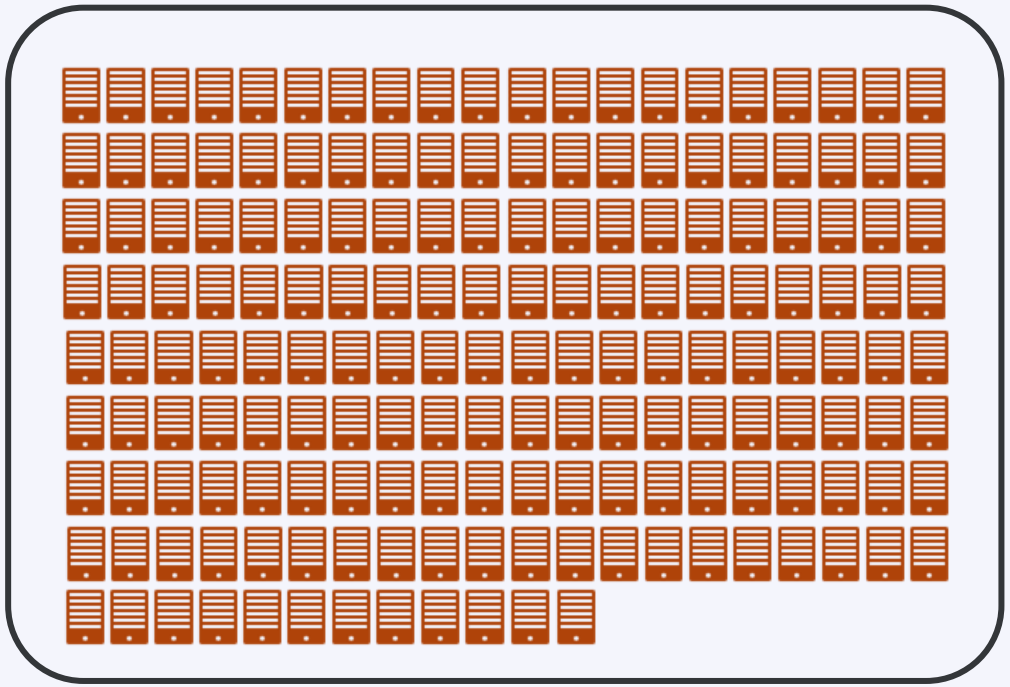
Tier-2 Clusters @ KU Leuven

Genius (since 2018)
250 nodes: 8,936 cores

wICE (since 9/2022)
186 nodes; 13,392 cores

Tier-2 Cluster - wICE

Compute nodes

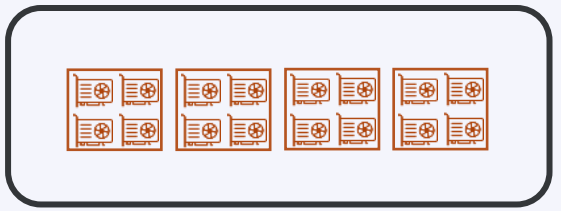


172x IceLake 72c 256 GB

Large memory nodes

5x IceLake 72c 2048 GB

GPU nodes



4x IceLake 72c 512 GB
4 A100 SXM4 80GB

Interactive nodes

5x IceLake 64c 512 GB
1 A100 80GB

Tier-2 Cluster - wICE

Type of node	CPU type	Inter-connect	# cores	installed mem	local discs	# nodes
Icelake	Xeon 8358	IB HDR-100	72	**256 GB**	960 GB	**172**
Icelake **large mem**	Xeon 8358	IB HDR-100	72	**2048 GB**	960 GB	**5**
Icelake **GPU**	Xeon 8358 **4xA100** SXM2 80GB	IB HDR-100	72	**512 GB**	960 GB	**4**
Icelake **Interactive**	Xeon 8358 **1xA100** SXM2 80GB	IB HDR-100	64	**512 GB**	960 GB	**5**

Storage

Overview of the storage infrastructure

- Only you own your files (POSIX)
  Users can share folders via VSC groups
- A VSC account has 3 default storages (free of charge)
  - `$VSC_HOME`
  - `$VSC_DATA`
  - `$VSC_SCRATCH`
- You can additionally request `staging` storage
- Different storage volumes have different:
  - mount point
  - size and performance
  - use case
  - backup and maintenance policy
- More info on ICTS Service Catalog (EN/NL)

Storage

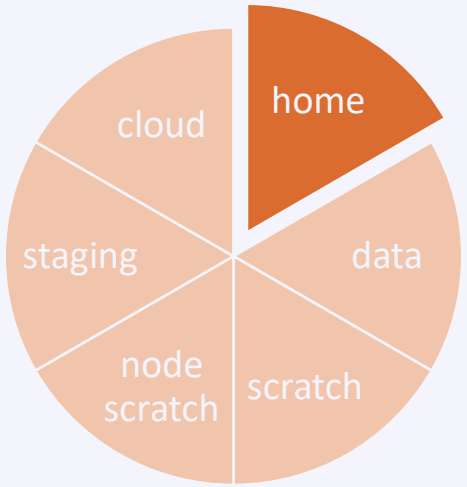
- **Do NOT use `/tmp`**
  It is only 10 GB and is reserved for the OS and root processes.
  Your application can crash if using `/tmp`
- You are automatically logged into your home folder upon login.
  Immediately go to your other storages, e.g.
  `$ cd $VSC_DATA`
- Always check your storage balance using `myquota` command

Example

```
$ myquota
file system $VSC_HOME
    Blocks: 1479M of 3072M
    Files: 12934 of 100k
file system $VSC_DATA
    Blocks: 12G of 75G
    Files: 1043k of 10000k
file system $VSC_SCRATCH
    Blocks: 15M of 1.5T
```

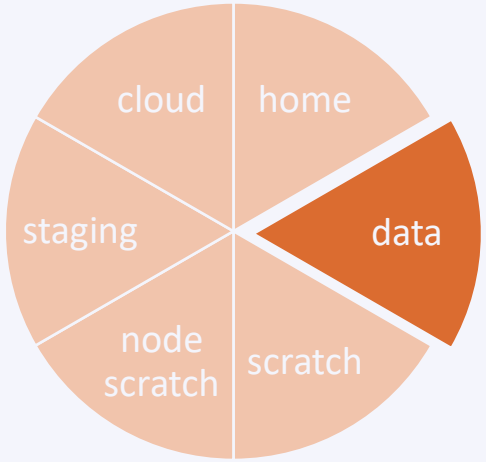
- Request form for extra storage
- More information

Storage



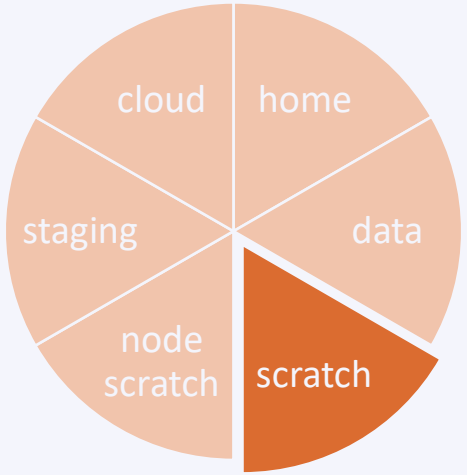
Storage	home folder
Env. Variable	`$VSC_HOME`
Filesystem Type	NFS
Access	Global
Backup	Hourly, daily, weekly (until last month) inside the `.snapshot` folder.
Default Quota	3 GB
Extension	Not possible
Usage	Only storing SSH keys, config files
Remarks	- Stay away from using it - Can easily overflow: + Your jobs may crash + Login issues

Storage



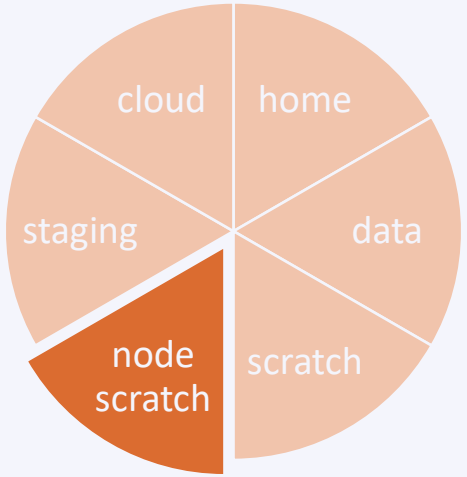
Storage	data folder
Env. Variable	`$VSC_DATA`
Filesystem Type	NFS
Access	Global
Backup	Hourly, daily, weekly (until last month) inside the `.snapshot` folder.
Default Quota	75 GB
Extension	On purchase
Usage	Your data, code, software, results
Remarks	- Permanent storage for initial/final results - Not optimal for intensive or parallel I/O

Storage



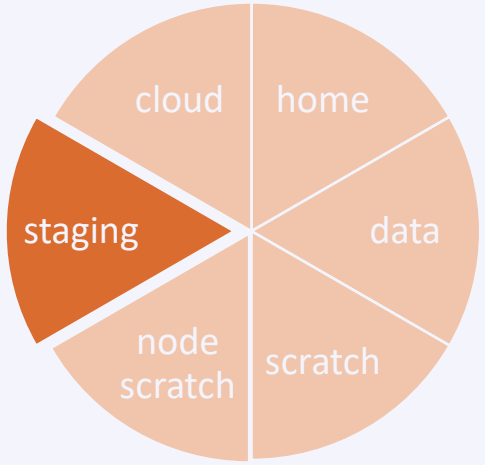
Storage	scratch folder
Env. Variable	`$VSC_SCRATCH`
Filesystem Type	Lustre
Access	Global
Backup	delete after 30 days from last access
Default Quota	500 GB
Extension	Free
Usage	Intensive, parallel I/O, temporary files
Remarks	- Recommended storage for all jobs - Copy scratch files to VSC_DATA or local storage after jobs are done - Deleted files cannot be recovered

Storage



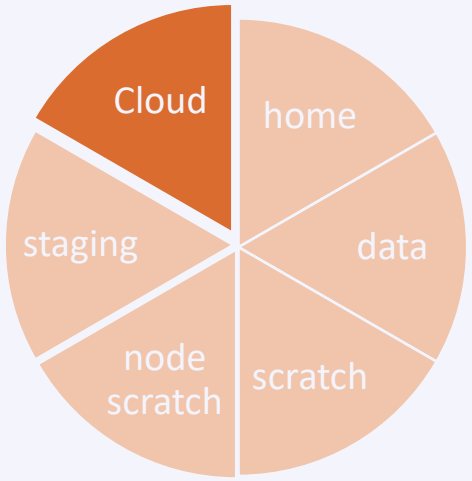
Storage	Node scratch folder
Env. Variable	`$VSC_SCRATCH_NODE`
Filesystem Type	Lustre
Access	On compute node, only at runtime
Backup	None
Default Quota	591 GB
Extension	Read about beeOND
Usage	Temporary storage at runtime
Remarks	- Fastest I/O, attached to the node - Is cleaned after job terminates - Copy the data to your home, scratch, or staging before job ends

Storage



Storage	Staging folder
Path	`/staging/leuven/stg_000XX`
Filesystem Type	Lustre
Access	On demand, only Tier-2@KUL
Backup	None
Default Quota	None
Extension	On purchase, from 1 TB
Usage	Permanent; share with a group
Remarks	- Accessible from login/compute nodes - Fast, parallel I/O

Storage



- VSC offers Tier-1 Data as a large storage service
- Only for active data (no archiving)
- Supports meta-data (search & discover)
- Share with the world (permission management)
- Free of charge (for academia)
- Starting volume is 1 PB
- Access is based on proposal
  Apply anytime

Credits

Credits card concept:

- Pre-authorization: holding the balance as unavailable until the merchant clears the transaction
- Balance to be held as unavailable: based on requested resourced (walltime, nodes)
- Actual charge based on what was really used: used walltime (you pay only what you use, e.g. when job crashes) See output file.

How to check the rates?	$ sam-quote
How to check available credits?	$ sam-balance
Did the new credits arrive?	$ sam-list-allocations

Credits Pricing

- You pay as you go!
- For academic projects:
  1000k credits = 3.5 EUR
- Credits needed for a job:

  #credits (k) = Walltime (hr) × #nodes × Factor

- For shared nodes, you pay a fraction of the costs, based on the --ntasks-per-node specified

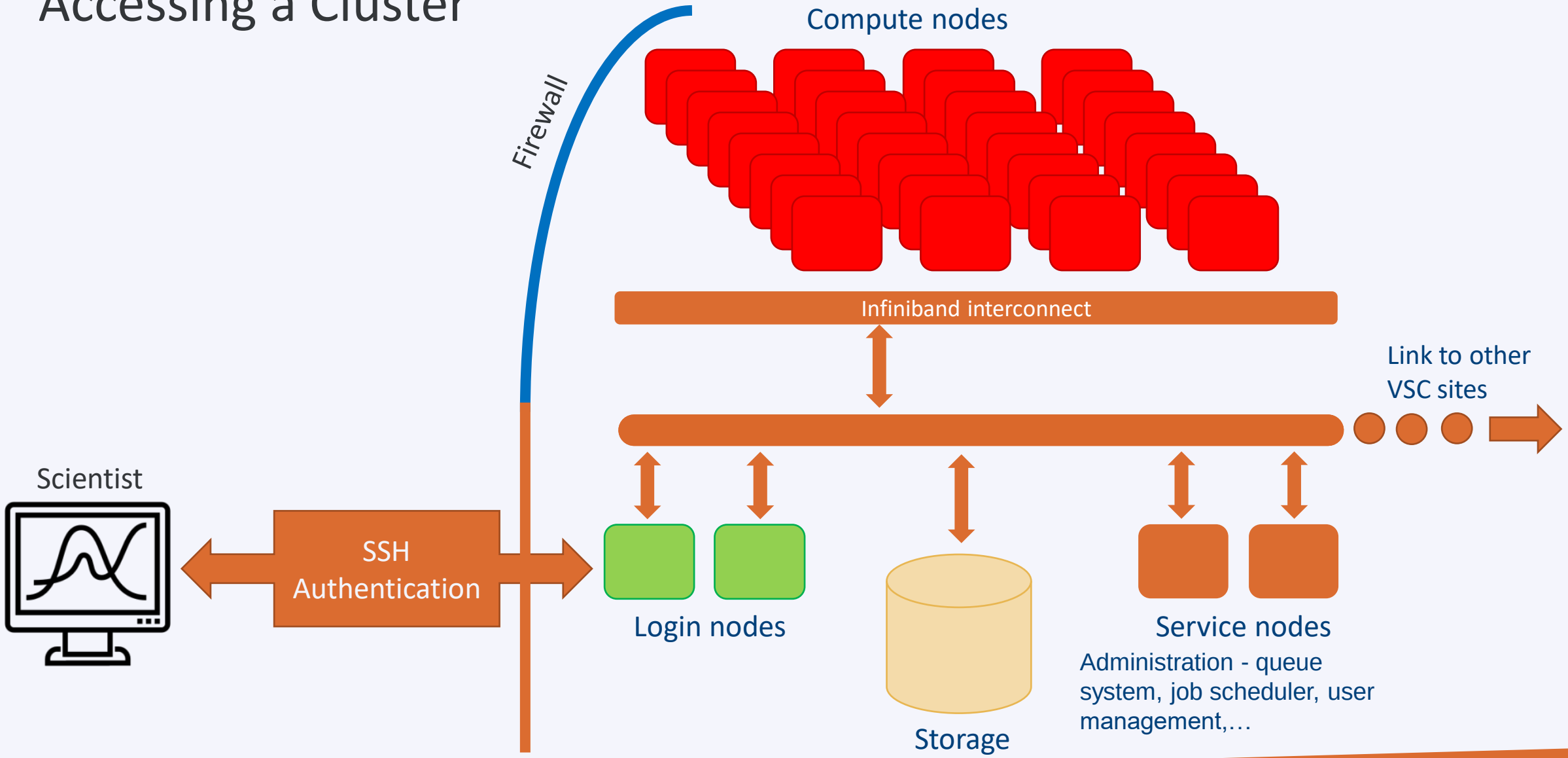
Cluster / Partition	Credits/hr
Genius Cascadelake	11.3
Genius Cascadelake 8 GPUs	40
Genius Skylake 4 GPUs	20
Genius Skylake BigMem	12
Genius Skylake	10
Genius / Superdome	10
Genius / AMD nodes	10
wICE / Thinnode	10,99
wICE Bigmem	19
wICE / 1 GPU	11,25
wICE 4GPUs	45
wICE Interactive	-

How to manage credits?

Command	Purpose
`sam-balance`	List active projects and available credits
`sam-list-allocations`	List the validity dates of different projects/allocations
`sam-list-usagerecords`	List current charge rates for different nodes
`sam-statement --account=<account-name>`	Job statements from an account

Login Nodes & MFA

Accessing a Cluster



How to login?

Open OnDemand

- Web-based login via browser
- https://ondemand.hpc.kuleuven.be
- Multi-Factor Authentication (MFA): Login first to your institute/university
- No SSH-key needed

NX

- Graphical desktop
- Using NoMachine client
- Using GUI (Matlab, SAS, visualization)
- Nvidia Quatro GPU for visualization

SSH client

- Windows: PuTTY, MobaXterm, NoMachine
  MacOS/Linux: terminal, NoMachine
- Requires (open)SSH keys
- Upload your key to VSC account page
  wait at least 30 minutes
- Hosts: login.hpc.kuleuven.be:22
  nx.hpc.kuleuven.be (NX)
- Use SSH agent (recommended)
  Use SSH Config (recommended)
- Can open GUIs if X11 is configured

Using Login Nodes

Tips

- To develop code
- To check your storage and credit balance
- To manage jobs (submit, check status, debug, resubmit, …)
- To move data around
  - within VSC: use data, scratch, staging
  - outside VSC: copy/sync from/to your local storage (e.g. Globus)
- To pre-/post-process your data/jobs
- To visualize your data
- To share files/folders

Warning

Do NOT execute heavy-lifting tasks (core, memory)

Login nodes are shared resources

Instead, submit jobs: e.g. Compile your software on compute nodes.

Check `$ slurmtop` to see how busy the cluster is

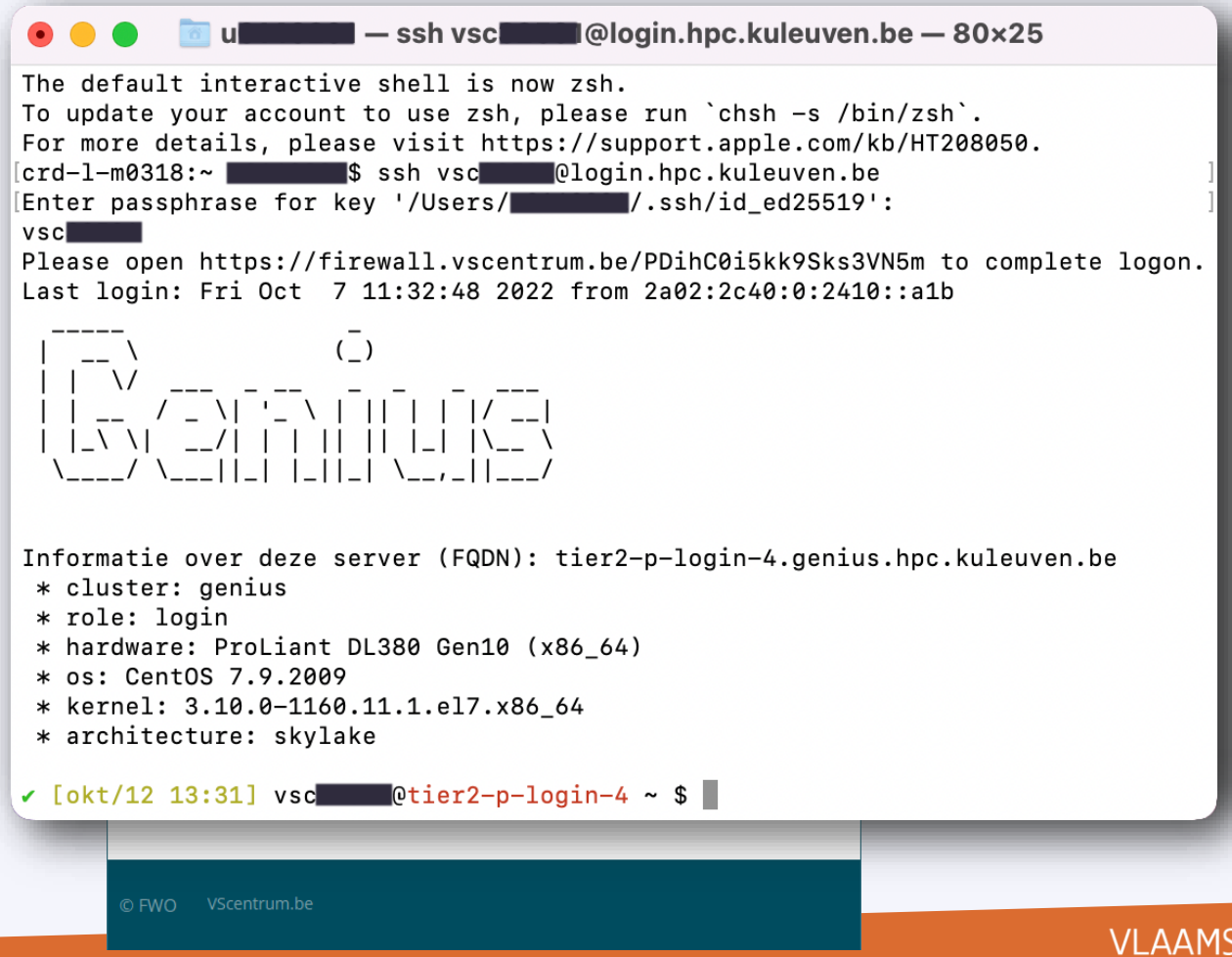
Connecting via Terminal

(Linux and Mac)

- Use ssh to connect:
  `$ ssh vscXXXXX@<host_name>`
- If key not found:
  `$ ssh -i </path/to/keyfile> ...`

If asked for **password**, please stop connecting and contact support, otherwise after a few attempts you will be blocked for 24h.

Host Name:
`login.hpc.kuleuven.be`



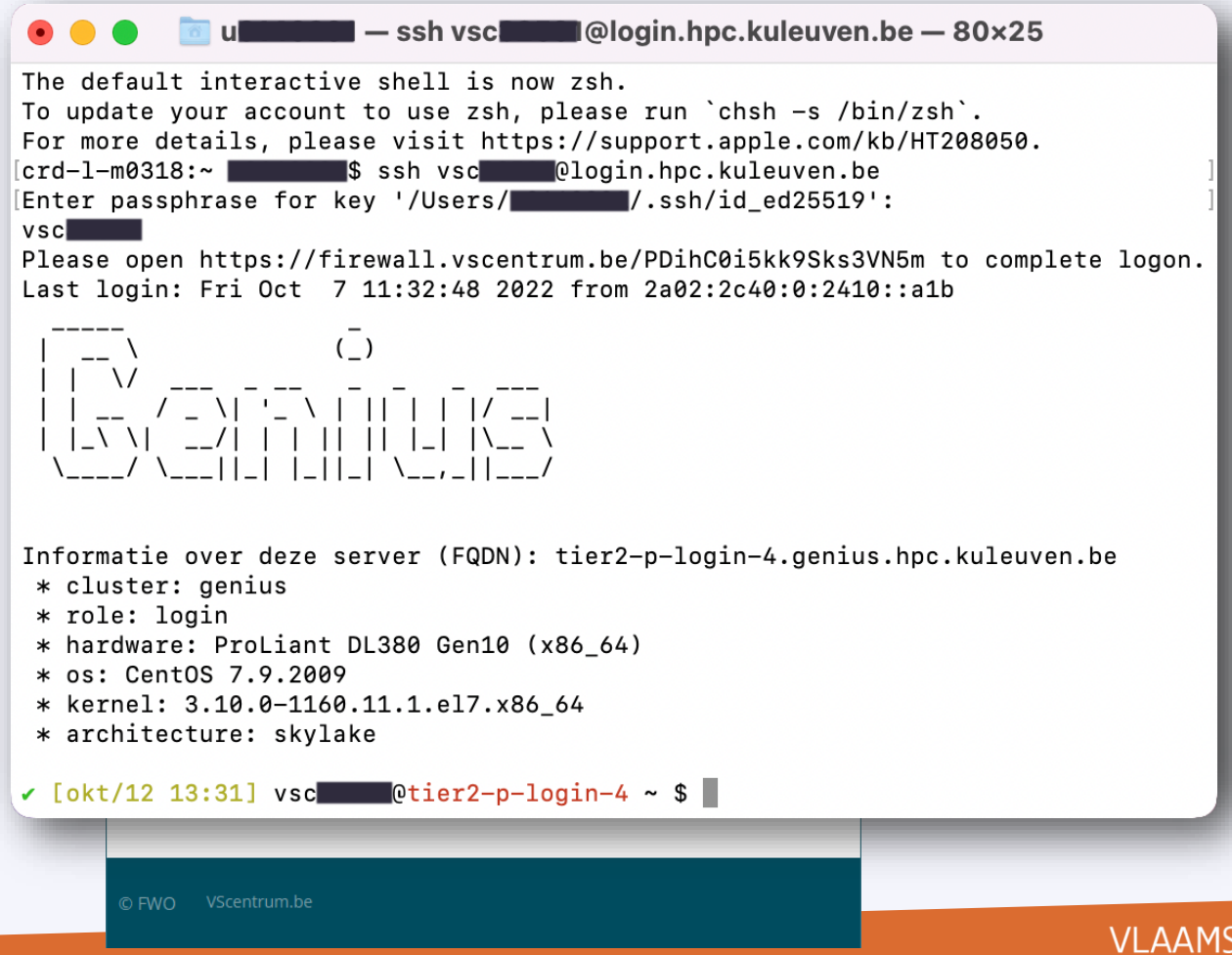
Connecting via Terminal

(Linux and Mac)

Host Name:
`login.hpc.kuleuven.be`

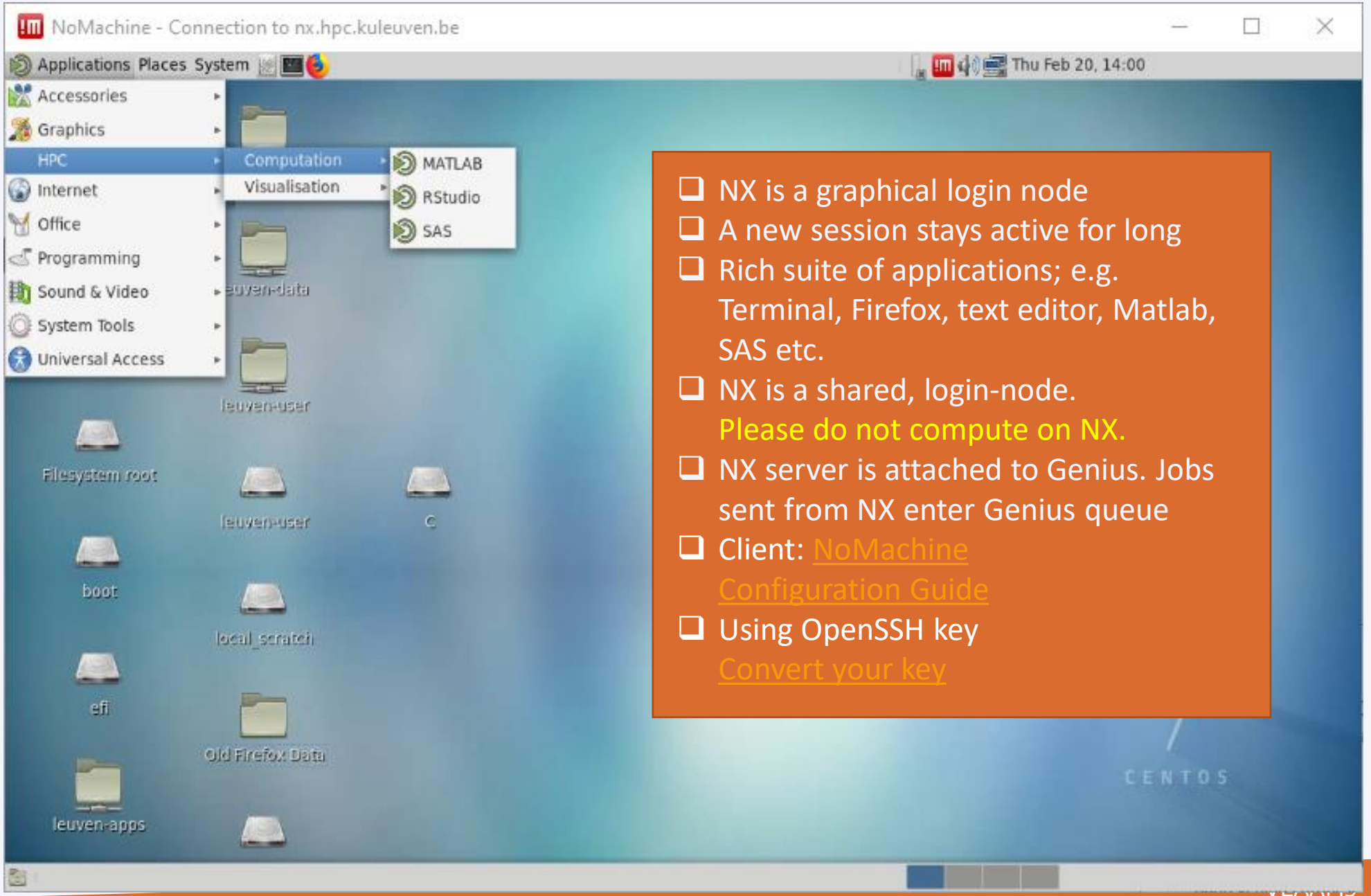
With SSH Agent

- Check your SSH Agent.
  Is your SSH key found?
  `$ ssh-add -l`
- If your SSH Agent is not running:
  `eval $(ssh-agent)`
- If your key is not found,
  add it to the Agent:
  `$ ssh-add </path/to/keyfile>`
- Use ssh to connect:
  `$ ssh vscXXXXX@<hostname>`



VLAAMS SUPERCOMPUTER CENTRUM

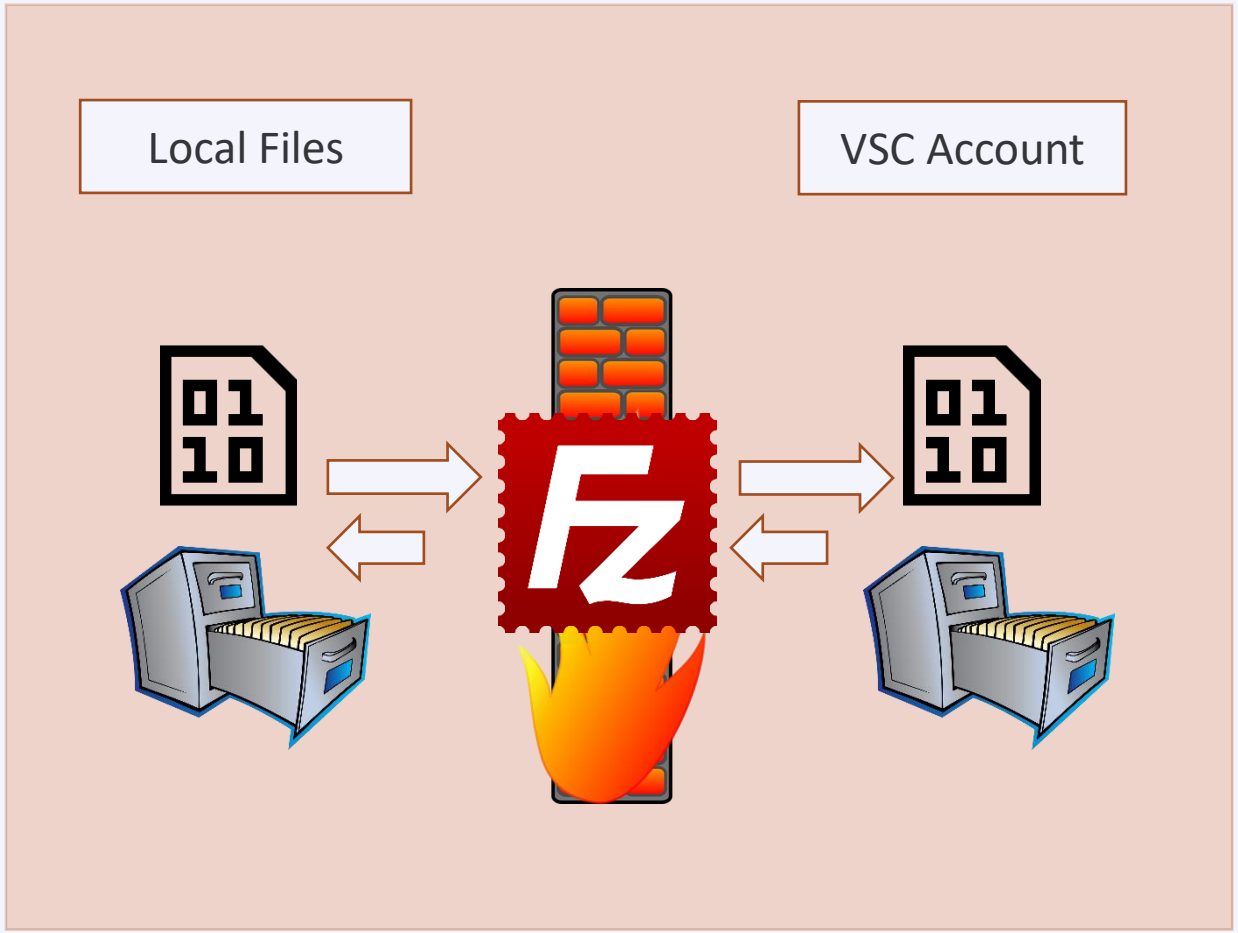
NX



- NX is a graphical login node
- A new session stays active for long
- Rich suite of applications; e.g. Terminal, Firefox, text editor, Matlab, SAS etc.
- NX is a shared, login-node. Please do not compute on NX.
- NX server is attached to Genius. Jobs sent from NX enter Genius queue
- Client: NoMachine Configuration Guide
- Using OpenSSH key Convert your key

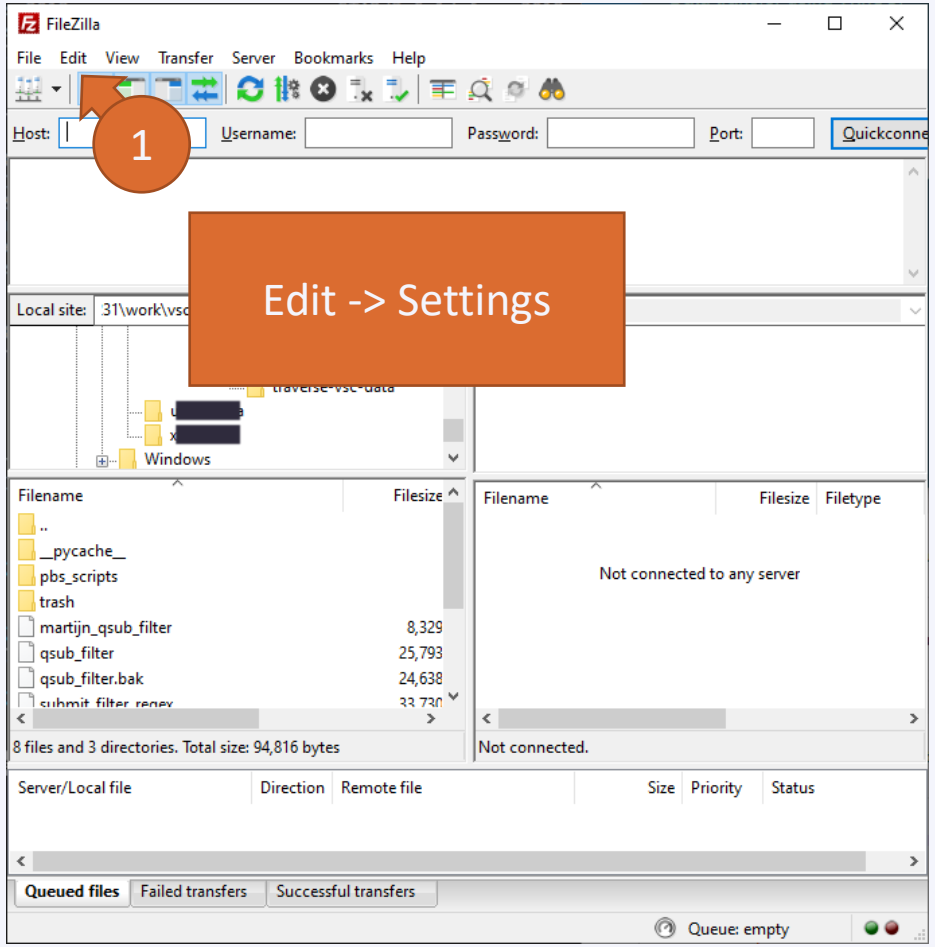
File Transfer with FileZilla

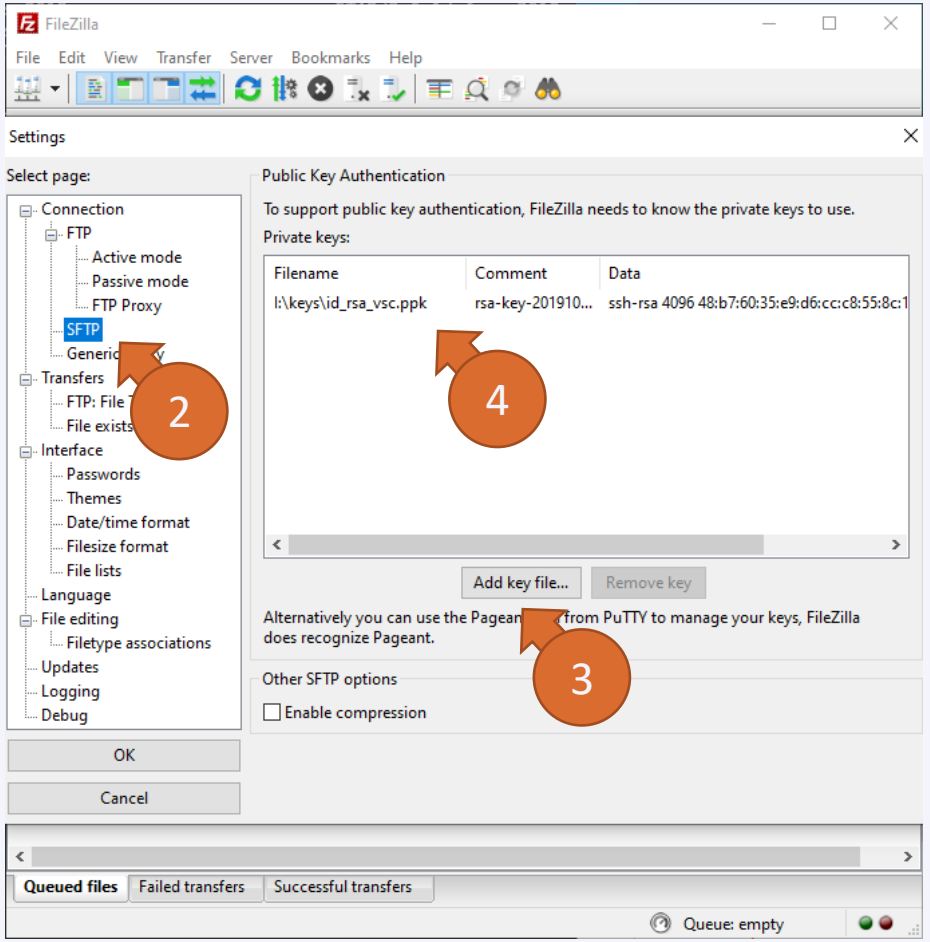
File transfer



Application	OS
FileZilla	Windows, Linux, Mac
WinSCP	Windows
rsync, scp	Linux, Mac
Globus	Window, Linux, Mac

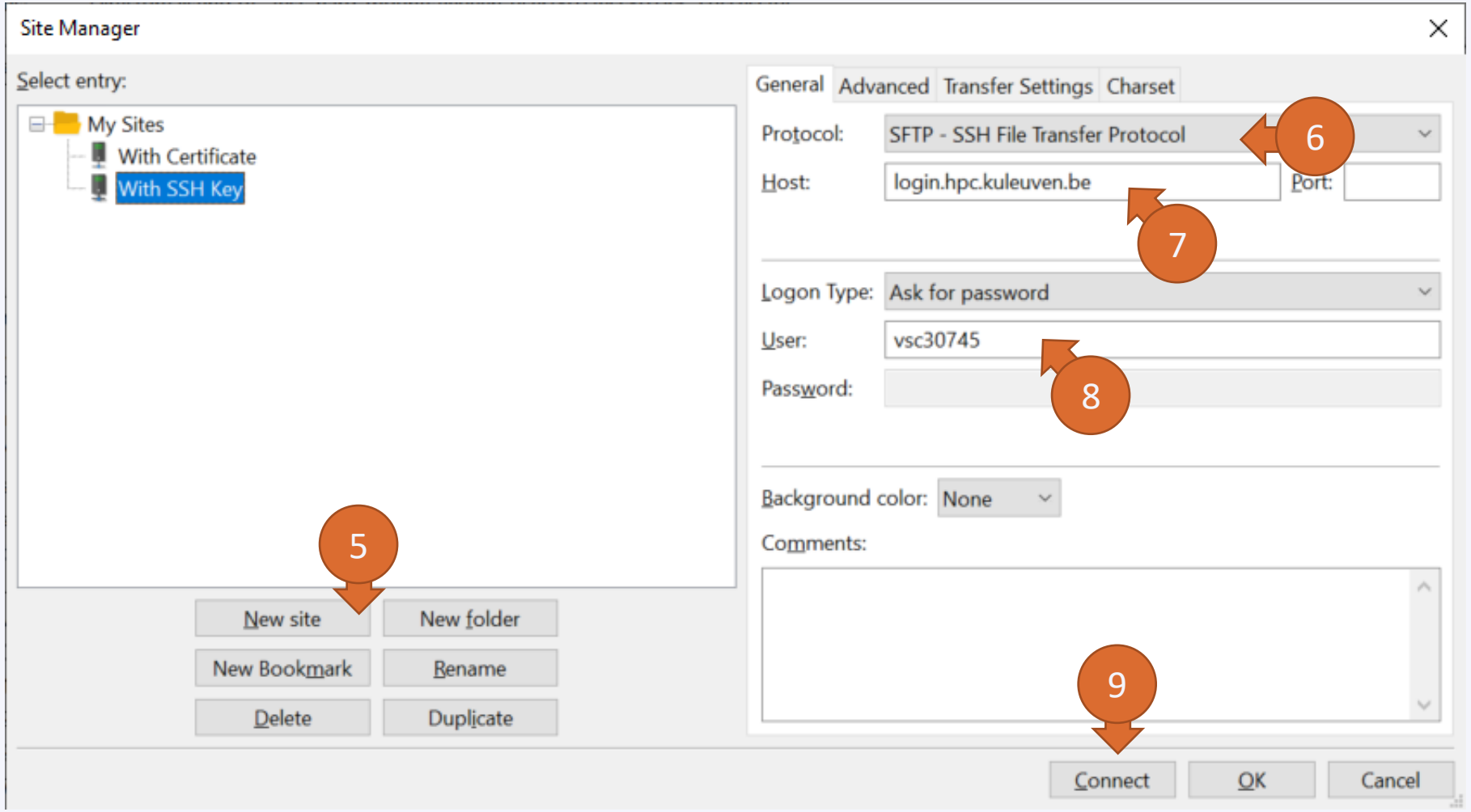
File transfer

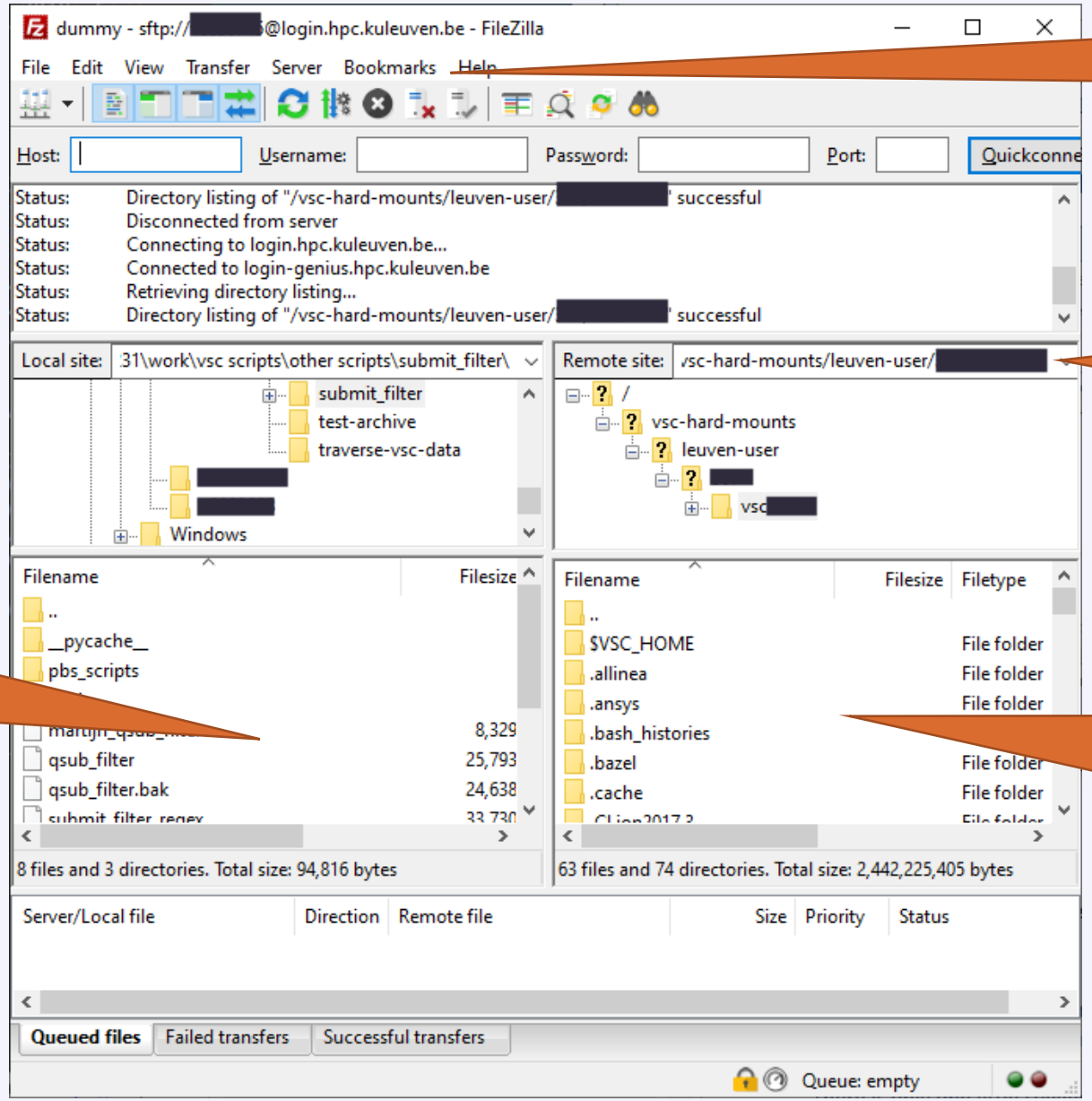




VLAAMS SUPERCOMPUTER CENTRUM

File transfer





For convenience, you'd better bookmark your data and scratch folders!

Instead, go to your `$VSC_DATA` folder, e.g. /data/leuven/399/vsc39934

Your local machine (e.g. laptop)

Your VSC storage (e.g. `$VSC_DATA`, `$VSC_SCRATCH`). **Note:** by default you arrive at your own `$VSC_HOME`. Copy nothing there!

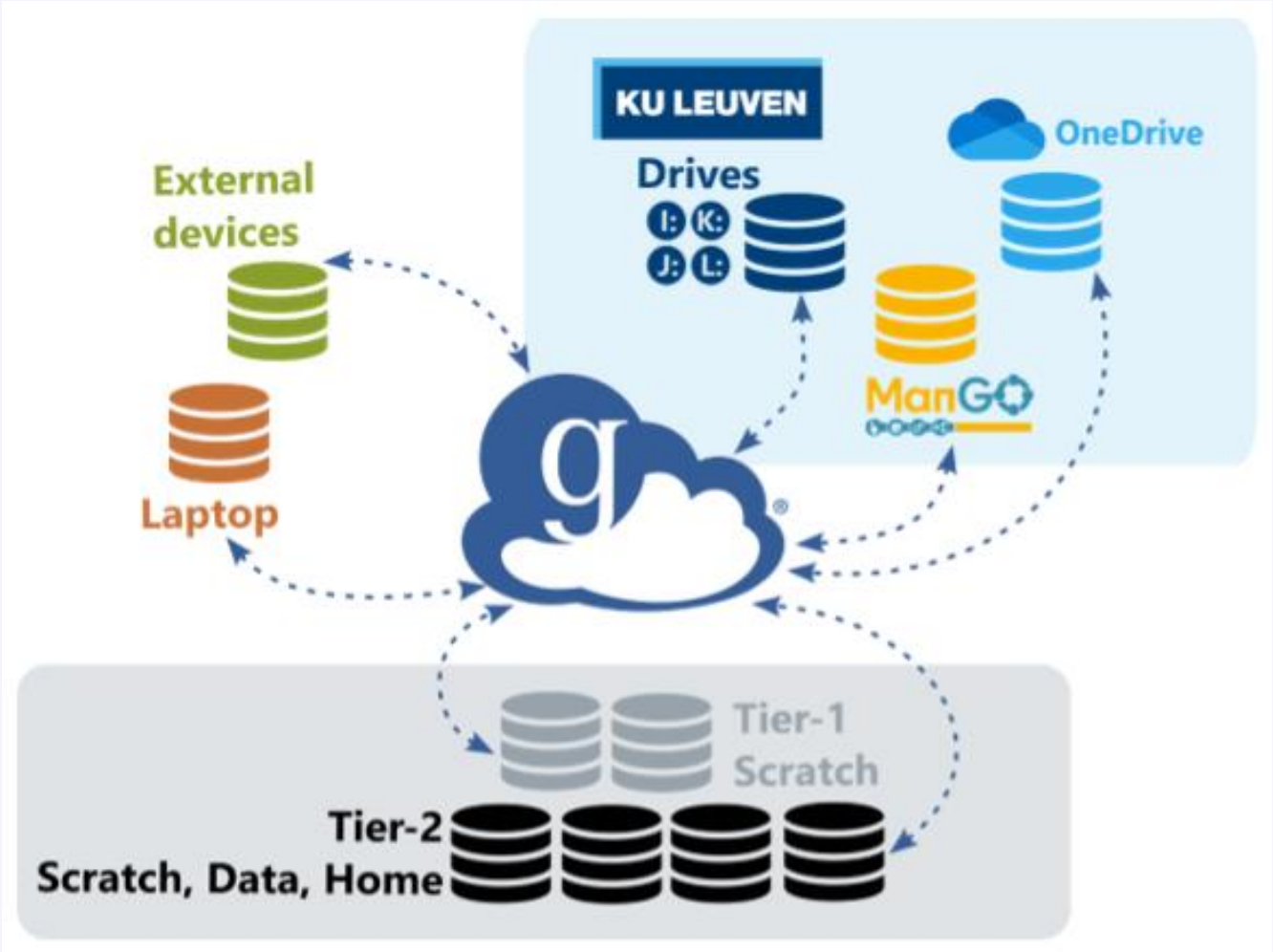
rsync

- For Linux, Mac and Windows WSL users
- Synchronize source and destination
- Requires SSH key
- Listens to your SSH agent
- Syntax

```
$ rsync -av /path/to/source vsc3XXXX@login.hpc.kuleuven.be:/path/to/destination
```

Globus

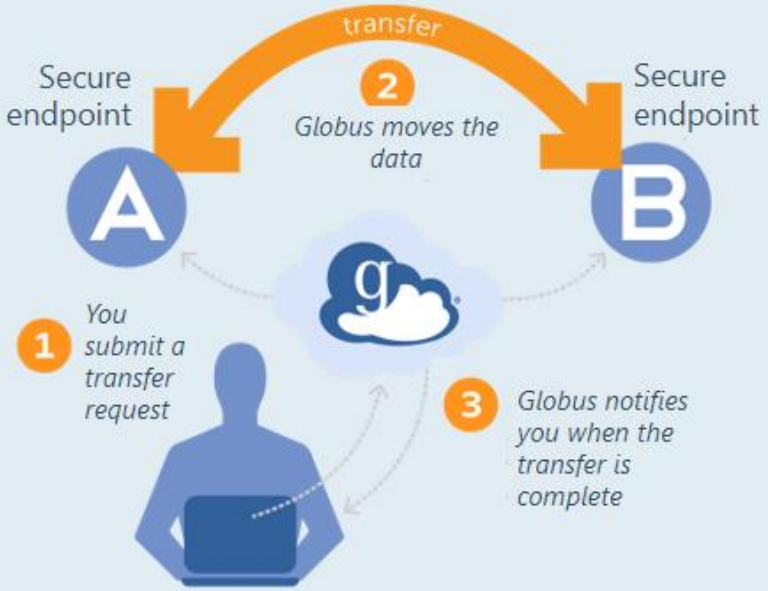
- Transfer, share and search within data
- Schedule and resume (large) transfers
- Fully supported by all VSC sites (available on Tier-2 and Tier-1)
- Web-interface
- Workflow:
  Create “endpoint” on the source
  Create “endpoint” on the destination
  Transfer between endpoints
- Existing endpoints on Tier-2 home/data/scratch storages



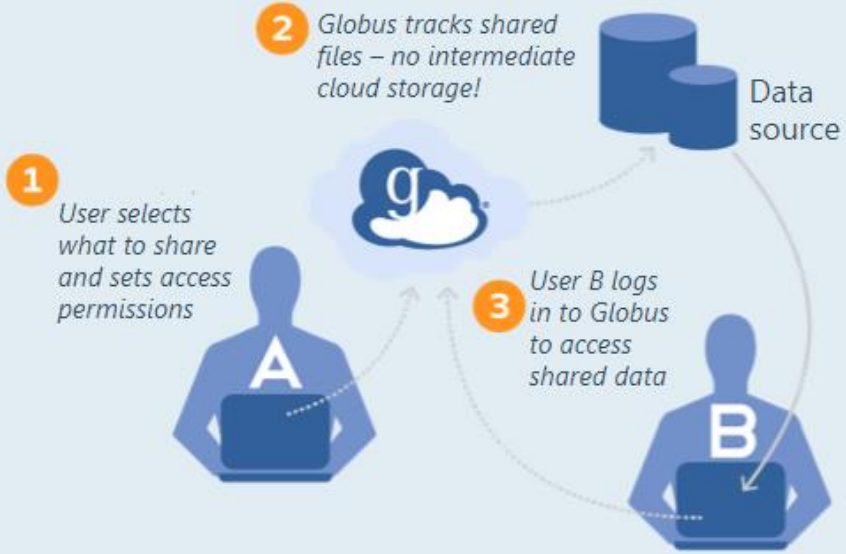
globus

globus.org

Transfer data



Share data

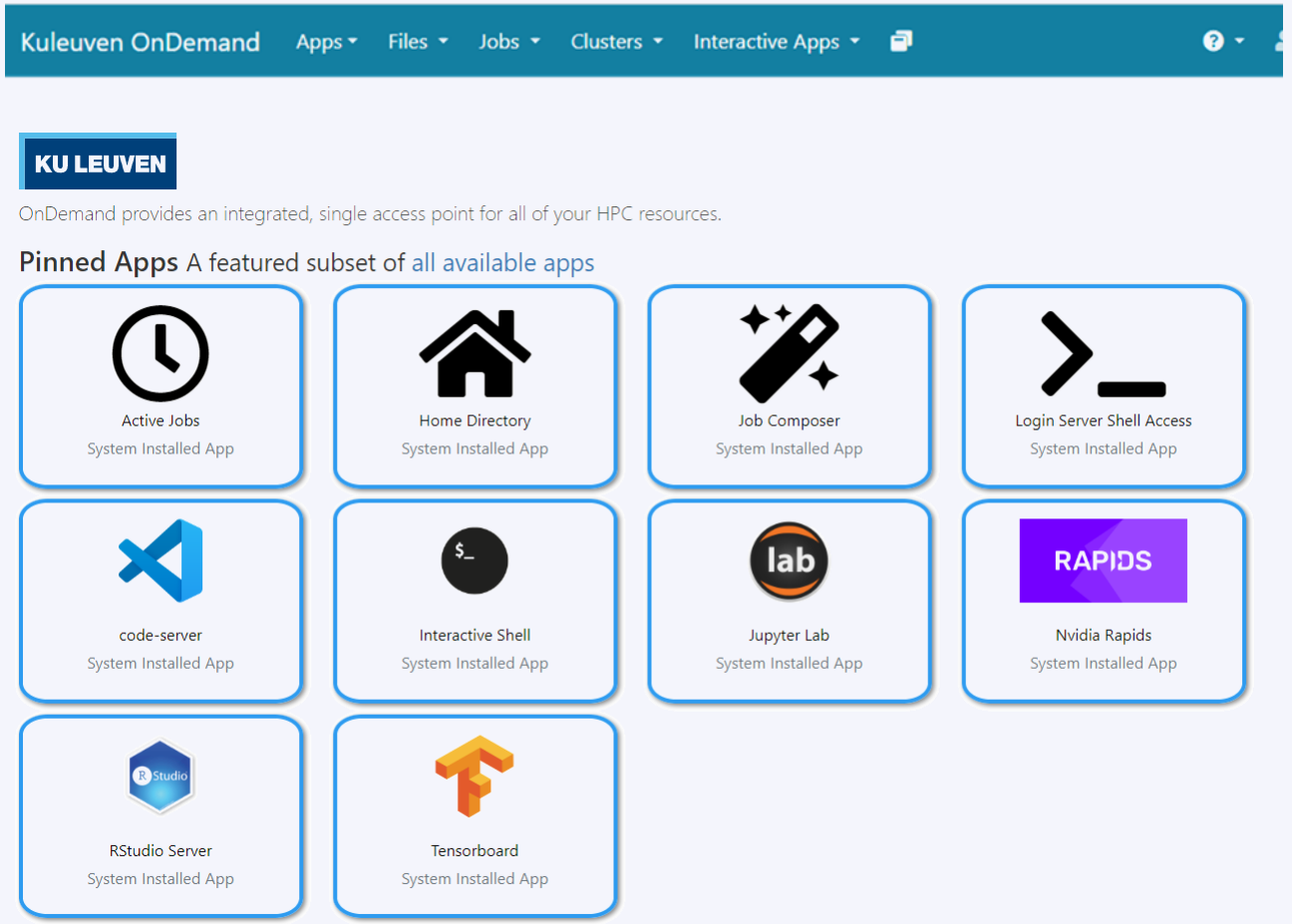


VLAAMS SUPERCOMPUTER CENTRUM

Open Ondemand

Open OnDemand

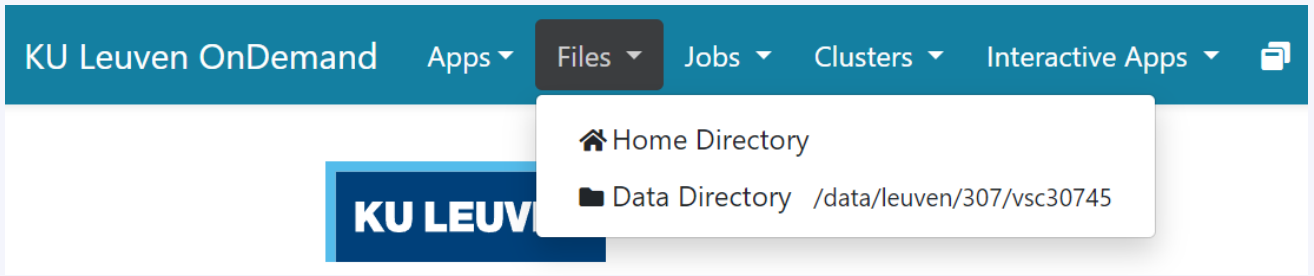
- Access clusters via web browser
- https://ondemand.hpc.kuleuven.be
- Login via KULeuven MFA
- File browser
- Interactive apps + integrated shell
- Create, start and monitor jobs
- Develop, compile and test your code
- VSCode editor
- Jupyter Lab, RStudio and Tensorboard



Open OnDemand

Access your VSC storage

- Use “Files” menu to access your VSC_HOME and VSC_DATA
- VSC_SCRATCH is not available

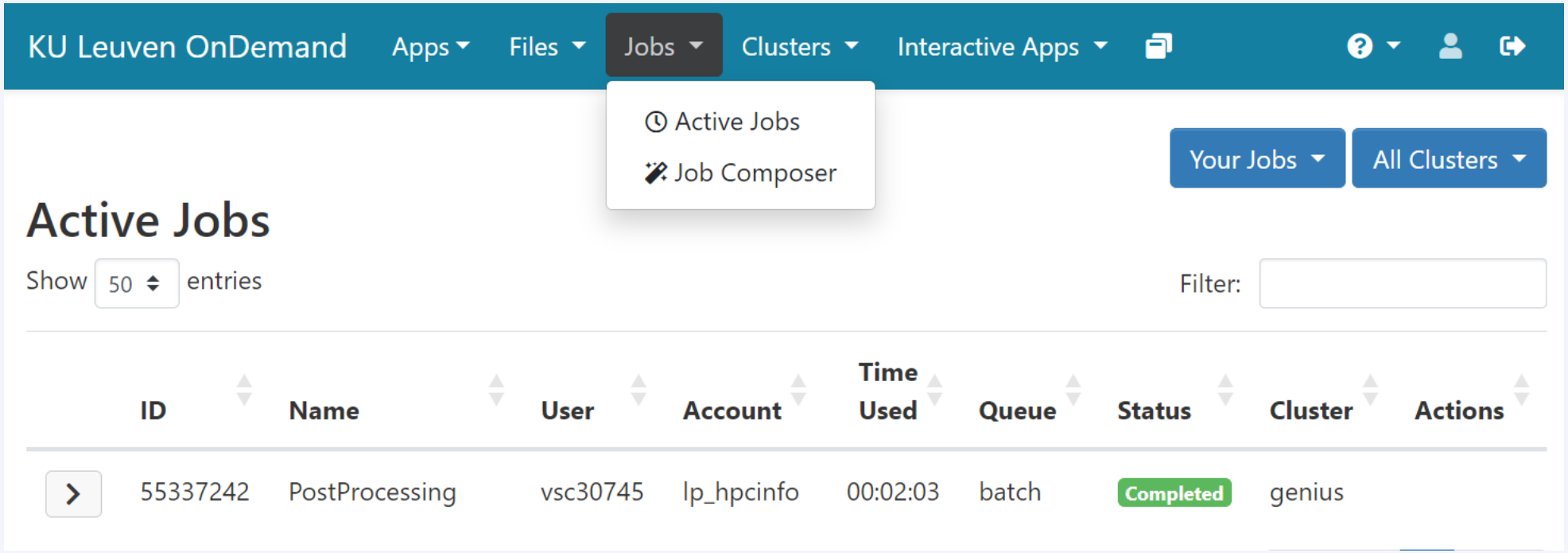


File/Folder Management

- OK to transfer small files/folders
- Open any (sub)folder in terminal (new window)
- Deleted files/folders could be retrieved from (hourly, daily, weekly) snapshots

Open OnDemand

Monitor/manage all your jobs

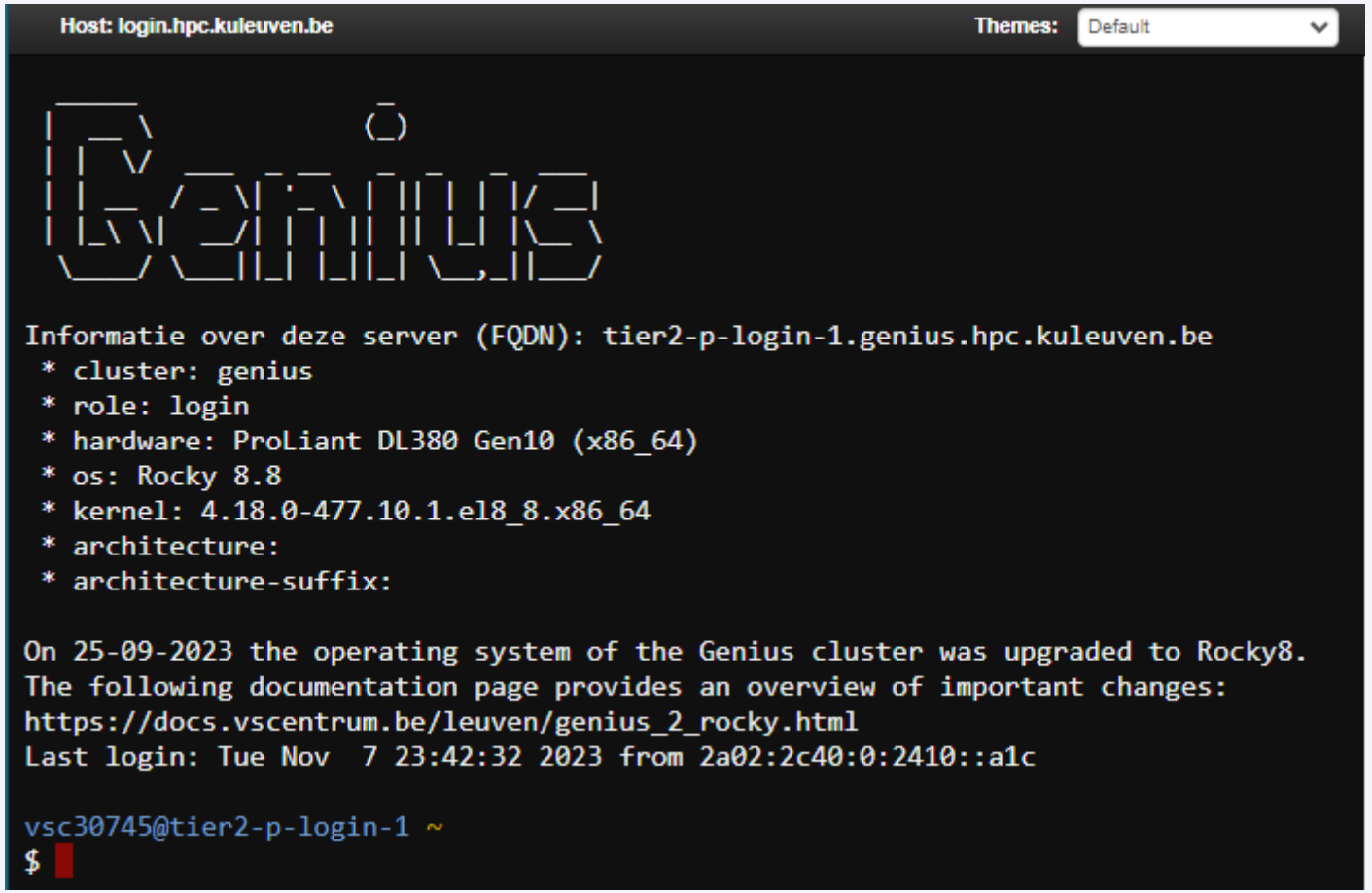


VLAAMS SUPERCOMPUTER CENTRUM

Open OnDemand: Login Server Shell Access

Login Shell

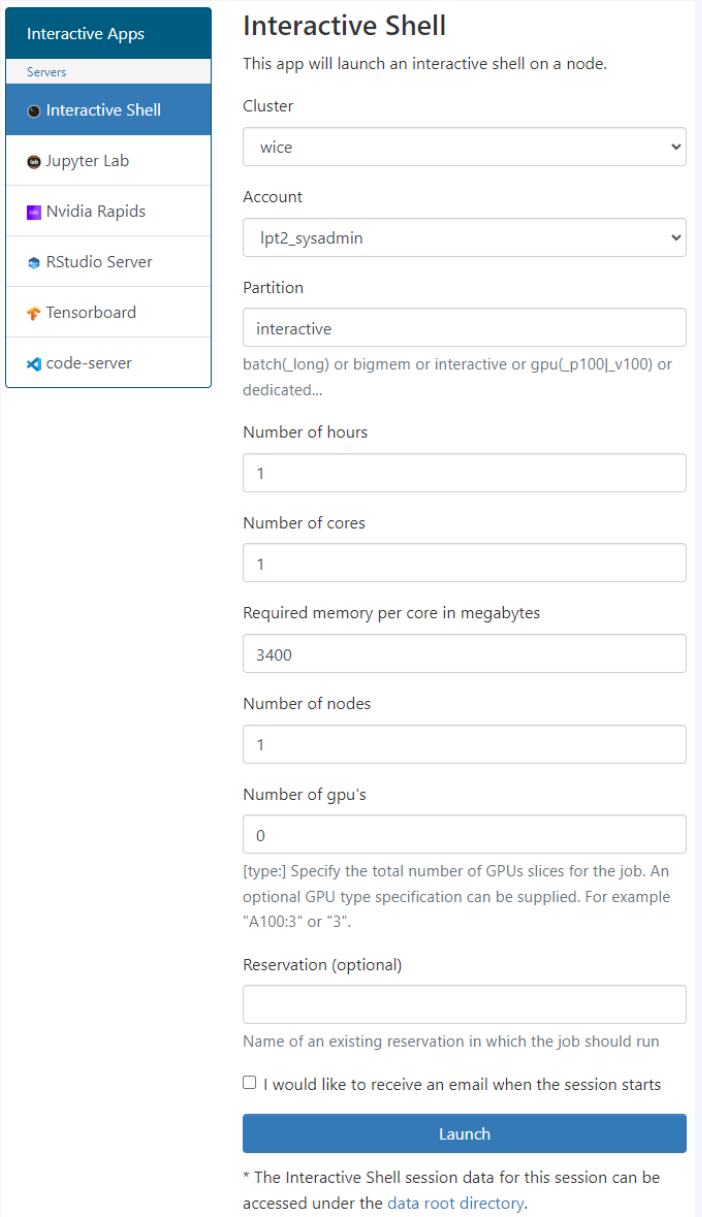
- Start shell in a new browser tab
- You land on Genius
  Accessing wICE goes via Genius login
- You land on your VSC_HOME
  `cd` to data/scratch/staging immediately
- Do NOT compute on the login nodes
- Try:
  ```
  $ sam-balance
  $ myquota
  ```



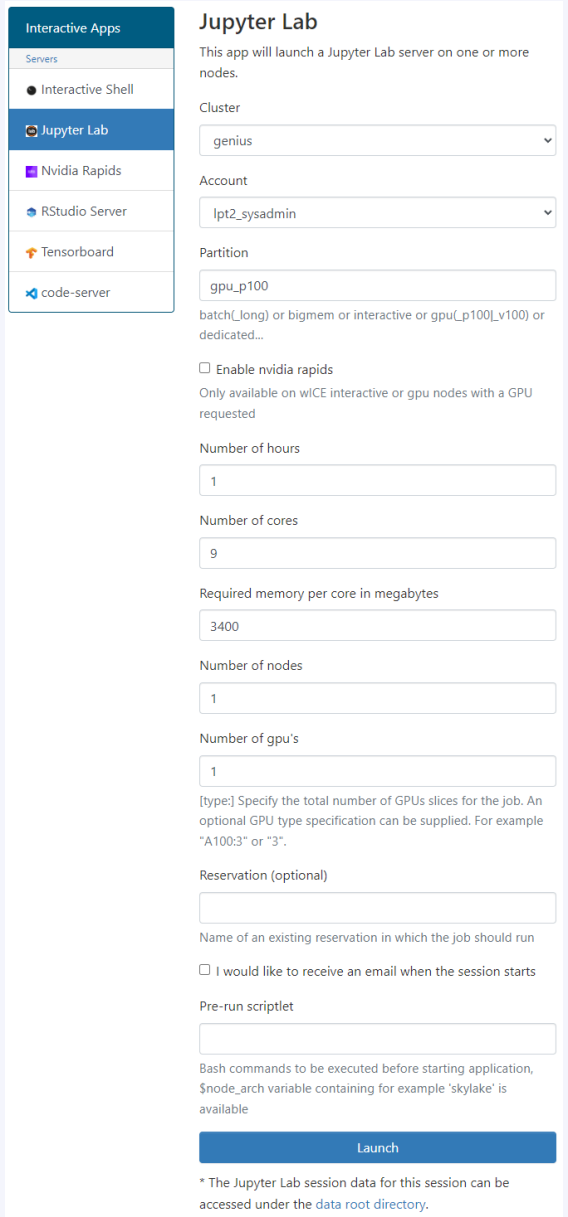
Open OnDemand: Interactive Shell

Interactive Shell

- Start shell as a job (on a compute node)
- Recommended to:
  Install your own software (in VSC_DATA)
  Test/run your program interactively
  Debug a program
- Choose the relevant cluster, partition and resources (core, memory, GPU)



Open OnDemand: Jupyter Lab



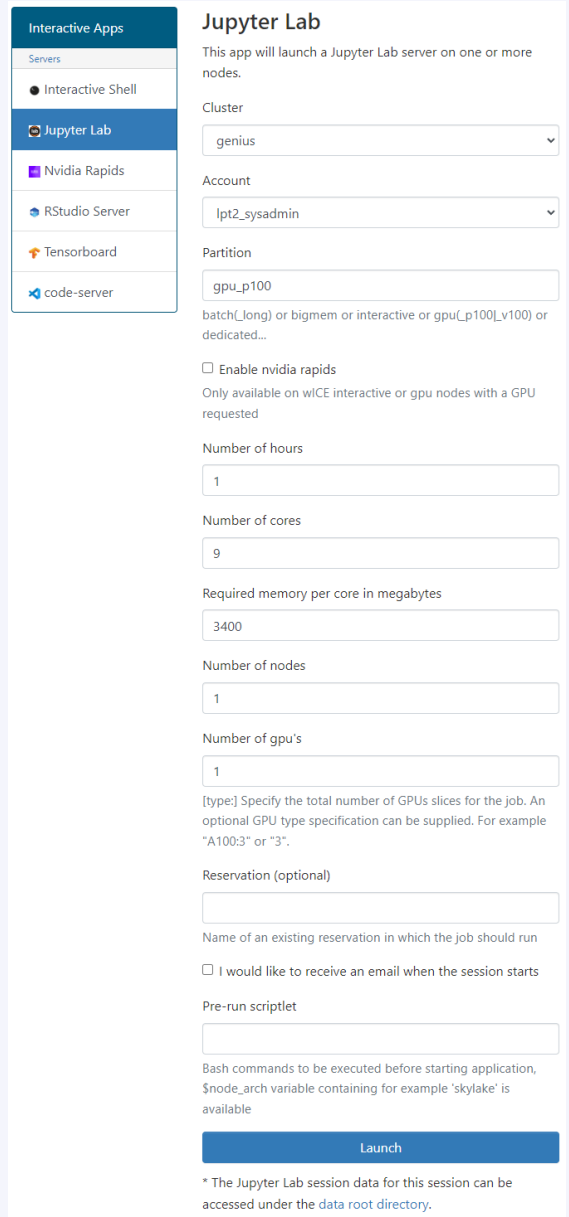
Jupyter Lab

- Create a notebook as a job (on a compute node)
- Best practice to:
  pre/post-processing your data
  prototyping
  add figures and text for pedagogical purposes
- Choose the relevant cluster, partition and resources (core, memory, GPU)
- User your own Python/R kernels (see next slide)

Open OnDemand: Jupyter Lab

Jupyter Lab

- Create a notebook as a job (on a compute node)
- Best practice to:
  pre/post-processing your data
  prototyping
  add figures and text for pedagogical purposes
- Choose the relevant cluster, partition and resources (core, memory, GPU)
- Extensions are not supported (for now)
- User your own Python/R kernels (see next slide)



Open OnDemand: Custom R/Python Kernels

My Kernels

- Default R/Python kernels are old and limited
- You can create a miniconda env and add it to your OnDemand kernels

Step 1: Miniconda

Start an Interactive shell

Go to your data folder

```
$ cd ${VSC_DATA}
```

Download miniconda

```
$ wget https://repo.continuum.io/miniconda/Miniconda3-latest-Linux-x86_64.sh
```

Install miniconda into your data folder

```
$ bash Miniconda3-latest-Linux-x86_64.sh -p $VSC_DATA/miniconda3
```

Permanenly add the path to miniconda to your `~/.bashrc`

```
$ echo 'export PATH="${VSC_DATA}/miniconda3/bin:${PATH}" ' >> ~/.bashrc
```

Open OnDemand: Custom R/Python Kernels

Step 2: New Environment

Create a new environment <env_name> with all packages you need, e.g.

```
$ conda create --name=<env_name> numpy scipy …
```

Enable your new environment

```
$ source activate <env_name>
```

Add the ipykernel package

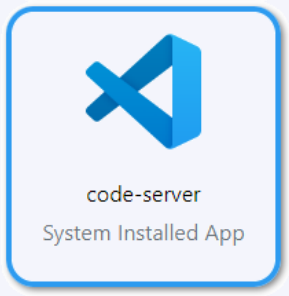
```
$ conda install -c conda-forge ipykernel
```

Step 3: Register Your Kernel

```
$ python -m ipykernel install -prefix=~/.local --name <env_name>
```

Start a new Jupyter Lab session, and the new kernels must be there

Open OnDemand: Other tools

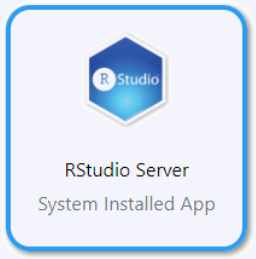


Start Visual Studio Code server as a job.

Develop, deploy and debug your workflow directly on HPC.

Using interactive partition on wICE is free of charge, and is relevant here.

Supports many languages + Github integration.



Start R IDE as a job.

Install your own packages in $VSC_DATA.

Track machine learning metrics and visualize data during the workflow.

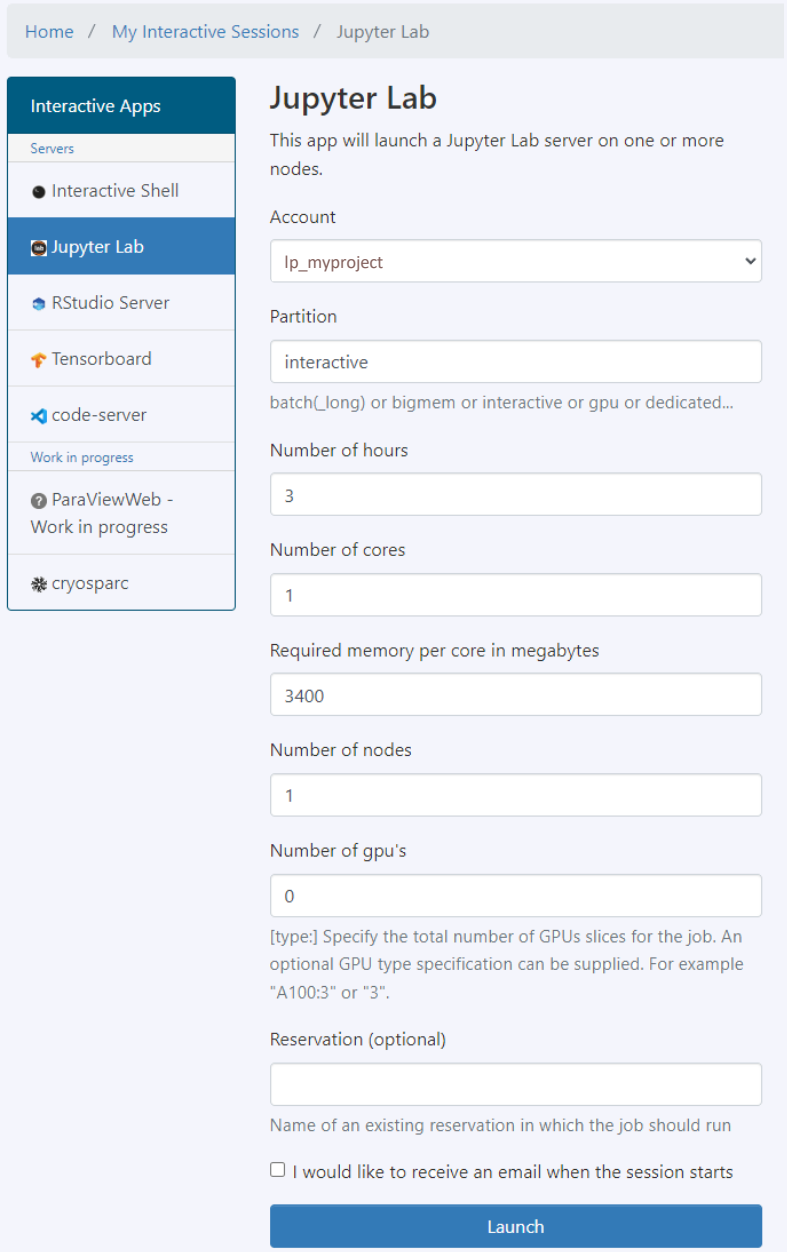
You need to provide a log folder.

Open OnDemand: Resources

E.g. to start a Jupyter Notebook

- Pick a valid Slurm credit account
- Default partition: batch
  interactive: Max 8 cores, 1 GPU (shard), 16 hr
- Default resources:
  1 core, 1 hour, 3400MB RAM, no GPU

(At the moment) you cannot use scratch and staging



Software Stack

Software: Available Modules

- OS: Rocky Linux 8.x
- Toolchains:
  - Intel (icc, icpc, ifort; Intel MPI; Intel MKL)
  - FOSS (gcc, g++, gfortran; OpenMPI; ScaLAPACK, OpenBLAS, FFTW)
  - Toolchain year on Genius from 2018a; on wICE from 2021a
- Note: **Never mix** FOSS and Intel compilers (gives dependency conflict)

Command	Remark
`module av`	List all available modules
`module av Python`	List all Python-related modules
`module spider Python`	Get more info
`module load Python/3.6.4-intel-2018a`	Load a specific module
`module list`	List all loaded modules and their dependencies
`module unload Python/3.6.4-intel-2018a`	Unload a module (but dependencies still stay)
`module purge`	Remove all modules from your work session

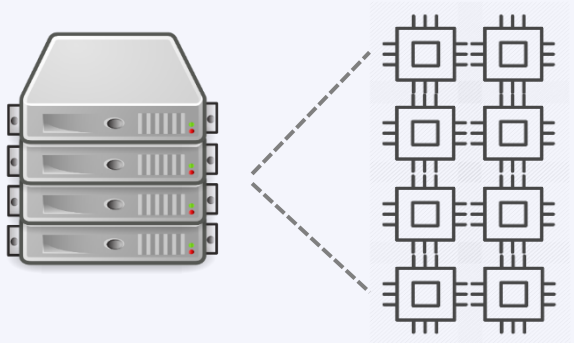
Software: Your Specific Needs

- You can always install your desired software in your $VSC_DATA
  Use Intel or FOSS toolchains
- Compile your code on a compute node (with interactive job)
- If you cannot, ask us for help
- Specific Python/R packages is managed by users via conda
- Read more about Python Package Management
- Read more about R Package Management

Starting to Compute

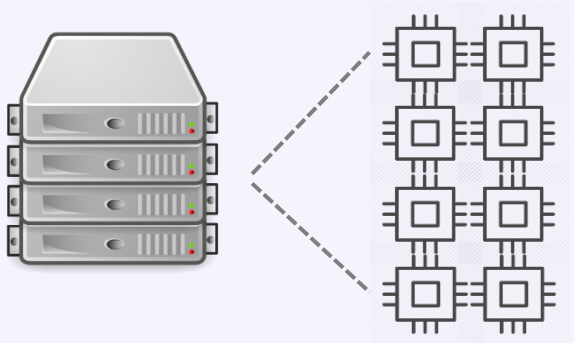
Resource Glossary

- **Cluster:** Which machine to use? Genius or wICE?
- **Nodes**: how many compute servers to request?
- **Cores**: how many cores per node to use?
- **Memory requirement**: how much memory each core needs?
- **Partition**: gpu, bigmem, superdome, amd
- **Walltime**: how long to use resources?
- **Storage**: how much storage (data, scratch, etc) the job needs?
- **Credits**: how many compute credits will be consumed?

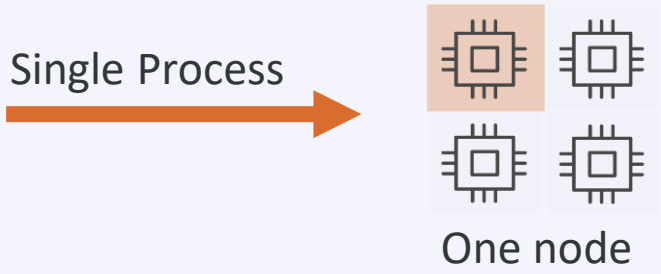


Default Resources

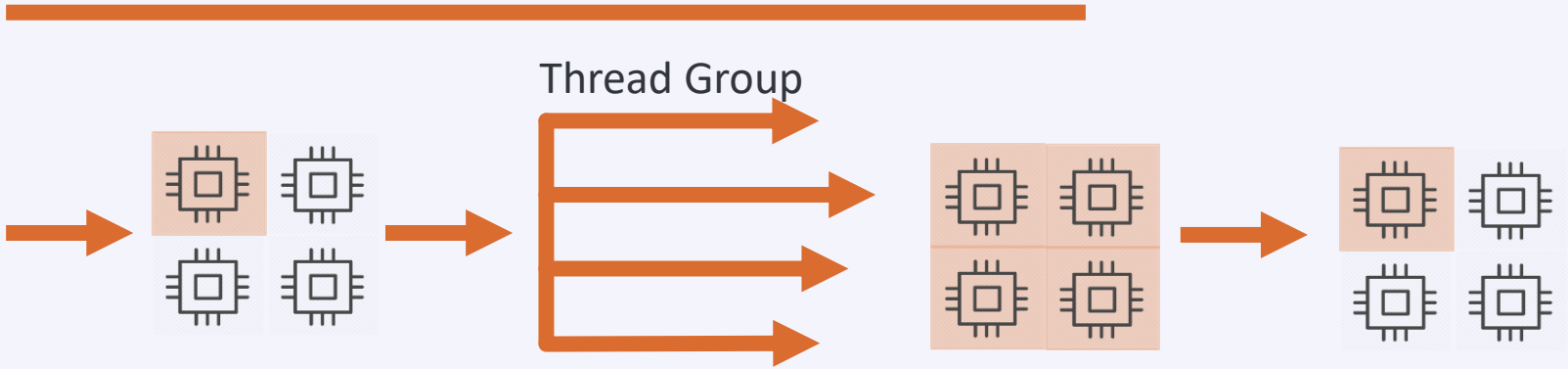
- **Cluster:** the local machine you are logged into (Genius or wICE)
- **Nodes:** 1
- **Cores:** 1
- **RAM:** depends on which node(s) you use
- **Partition:** batch
- **Walltime:** 1 hour
- **Storage:** no default
- **Credits:** no default



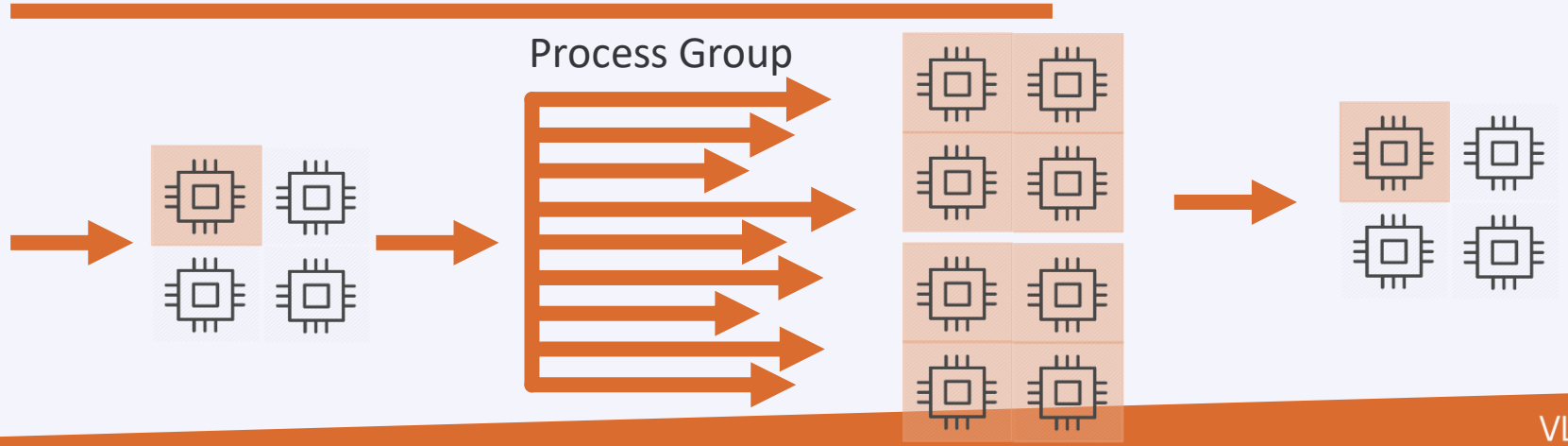
Serial Application
(1 process on 1 core)



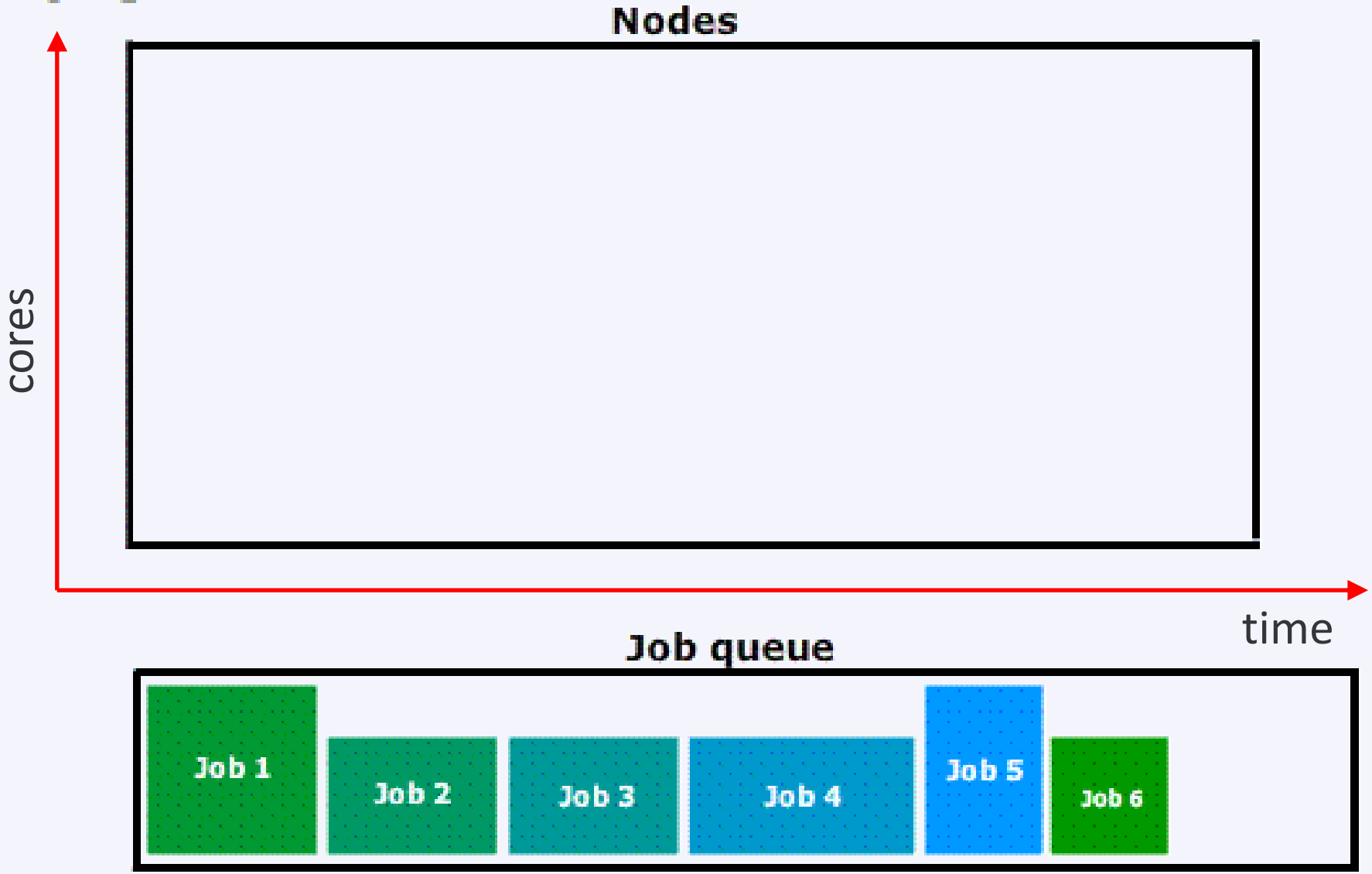
Multi-Core Appl.
(N threads on N cores from **1 node**)



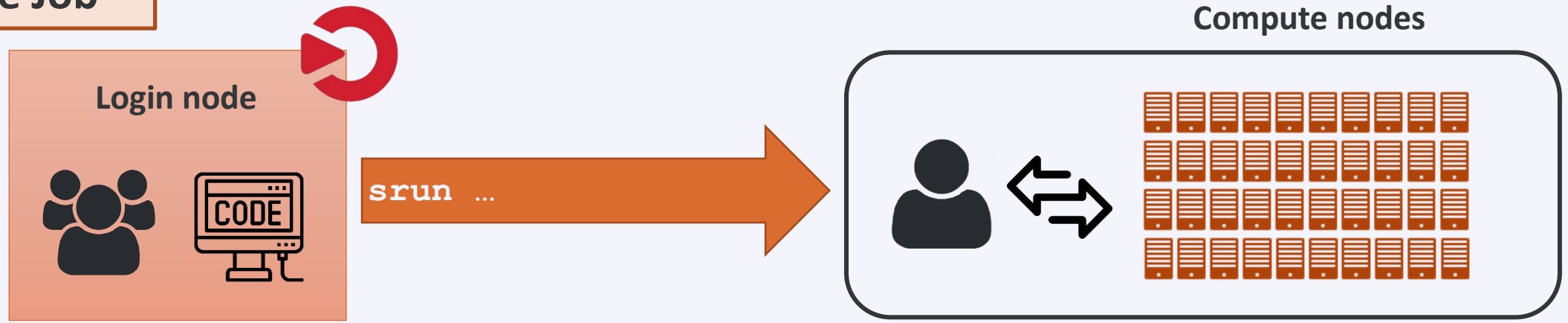
Distributed Appl.
(many processes on multiple cores/nodes)



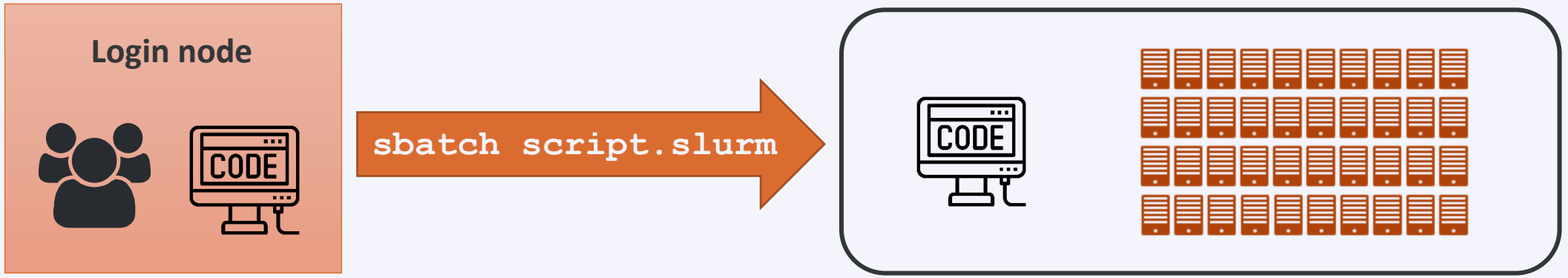
Backfill



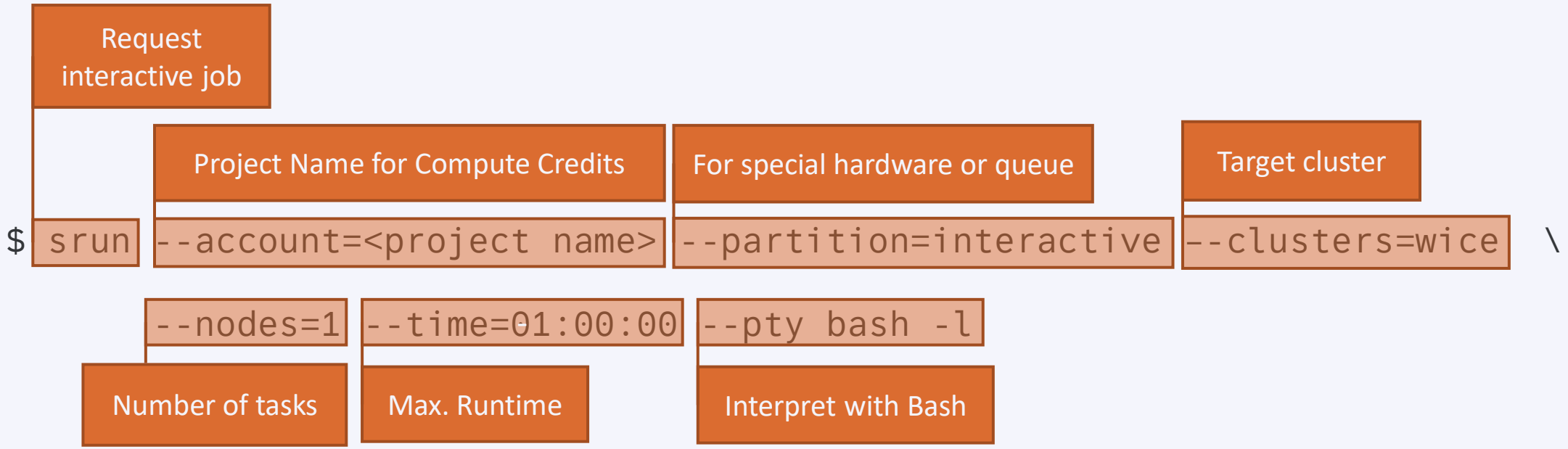
Interactive Job



Batch Job



Interactive Job on **wICE**



Remark

- Specifying <project name> for credits is mandatory, *e.g.*: **`-A lp_hpcinfo`**
- Implicit defaults for an interactive job on wICE are: `-n 1, -t 01:00:00, --mem-per-cpu=2000M`

Command	Purpose
`$ sbatch ...`	Submit a batch job
`$ srun ...`	Submit an interactive job
`$ scancel --cluster=wice <JobID>`	Cancel a specific pending or running job
`$ scontrol show job --clusters=wice <JobID>` `$ slurm_jobinfo <JobID>`	Detailed job info (very useful to diagnose issues)
`$ squeue --clusters=wice --long`	Status of all recent jobs
`$ squeue --clusters=wice --start`	Give a rough estimate of start time
`$ sinfo --clusters=wice`	Info about the state of available partitions and nodes
`$ sacct --clusters=wice --batch --job <JobID>`	Show minimal info about a queue or partition (-p)
`$ slurmtop` `$ scontrol --clusters=wice show node <hostname>`	Overview of the cluster Get detailed information about the status of a node
`$ sam-balance`	Overview of all your available credit projects
`$ sam-list-allocations`	Detailed overview of your credit allocation history

Jobs: Slurm submit options

SLURM	Remarks

`--nodes=X --ntasks-per-node=Y`
`--ntasks=X*Y`
`--ntasks=X --cpus-per-task=Y`
	or
	or
	(*e.g.* Hybrid MPI + OpenMP)

`--partition=<partition_name>`	Default: "batch" partition
`--mem-per-cpu=<XG>`	Minimum memory per core
`--time=<dd-hh:mm:ss>`	e.g. 1-12:30:00
`--job-name=<job_name>`	
`--output=<file_template>`	STDOUT; defaullt="slurm-%j.out"
`--error=<file_template>`	Default: redirect to STDOUT
`--mail-type=FAIL,BEGIN,END`	
`--mail-user=<email-address>`	
`--export=<ALL,key=value>`	Additional variables to pass to job
`--account=<account_name>`	Mandatory (credits)

Argument Shorthands

Some (not all) of the `sbatch`, `srun` and `salloc` command line arguments have shorthands

Shorthand	Full Argument	Meaning
`-A`	`--account`	Slurm account name
`-a`	`--array`	Job array range
`-M`	`--clusters`	Machine or cluster name
`-c`	`--cpus-per-task`	Num cores per task (default=1)
`-d`	`--dependency`	Job dependency (after, afterok, afterany, …)
`-I`	`--input`	STDIN filename
`-e`	`--error`	STDERR filename
`-o`	`--output`	STDOUT filename
`-t`	`--time`	Maximum walltime
`-N`	`--nodes`	Minimum num nodes
`-n`	`--ntasks`	Maximum num tasks
`-p`	`--partition`	Partition name

Interactive Jobs

- Interactive job: 1 core for 1 hour (default)

```
$ srun --clusters=wice --account=lp_hpcinfo --pty /bin/bash -l
```

- Interactive job with X-forwarding

```
$ srun --clusters=wice --account=lp_hpcinfo -x11 --pty /bin/bash -l
```

- Request fraction of a node from interactive partition

```
$ srun --clusters=wice --account=lp_hpcinfo --nodes=1 --ntasks=4 \
       --partition=interactive --pty /bin/bash -l
```

- Request a GPU accelerator

```
$ srun --clusters=wice --account=lp_hpcinfo --nodes=1 --ntasks=18 \
       --partition=gpu --gpus-per-node=1 --pty /bin/bash -l
```

Example Slurm Job Script

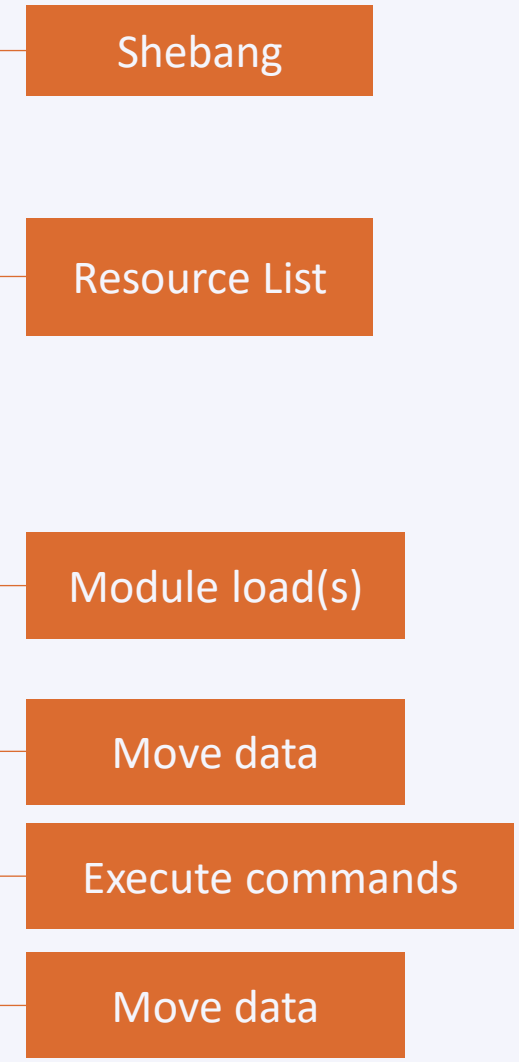
jobscript.slurm

```
#!/bin/bash -l
#SBATCH --clusters=wice
#SBATCH --account=lp_hpcinfo
#SBATCH --nodes=1
#SBATCH --ntasks=4
#SBATCH --mem-per-cpu=4G
#SBATCH --job-name=hpc_workflow
#SBATCH --mail-type=BEGIN,END,FAIL
#SBATCH --mail-user=my.name@kuleuven.be

module load intel/2021a
module load Python/3.9.5-GCCcore-10.3.0
which python3
cd $VSC_SCRATCH/projects/simulations
cp -r $VSC_DATA/input_data .

python modelling.py

cp -r output_data $VSC_DATA
rm -rf ./input_data ./output_data
```



Batch Workload

- Submit the job to the batch server
- Receive a unique JobID
- Error and output files

Job Script

```
#!/bin/bash -l
…
```

Command Line

```
$ sbatch simulation.slurm
```

JobID

```
Submitted batch job 60042478 on cluster wice
```

stderr, stdout

```
$ ls *.out
slurm-60042478.out
```

Output File

- STDERR and STDOUT are by default redirected to a single file:
- `slurm-<JobID>.out`
- Contains job info, all errors and warnings, and printouts
- Always study it
- Address all warnings and errors (if you can)
- Typical error examples …

STDOUT + STDERR

```
$ ls slurm-*.out
slurm-60042478.out
```

Out of Memory

```
slurmstepd: error: Detected 1 oom-kill event(s).
Some of your processes may have been killed by the
cgroup out-of-memory handler.
```

Short Walltime

```
slurmstepd: error: *** JOB 60042478 ON s28c11n2
CANCELLED AT 2023-02-08T10:03:43 DUE TO TIME LIMIT
***
```

Low Disk Space

```
IOError: [Errno 122] Disk quota exceeded
```

Output File

- Always created
- `slurm-<job_id>.out`
- Contains all standard output and error (instead of screen)
- Always study it
- Standard Output and Error channels can be redirected to other files:

```
#SBATCH --output …
#SBATCH --error …
```

stdout

```
$ ls slurm-*.out
slurm-60041238.out
```

Output File

```
SLURM_JOB_ID: 60033947
SLURM_JOB_USER: vscXXXXX
SLURM_JOB_ACCOUNT: lp_wice_pilot
SLURM_JOB_NAME: testjob
SLURM_CLUSTER_NAME: wice
SLURM_JOB_PARTITION: batch
SLURM_NNODES: 1
SLURM_NODELIST: m28c30n4
SLURM_JOB_CPUS_PER_NODE: 72
Date: Tue Jan 10 17:02:04 CET 2023
Walltime: 00-01:00:00
==================================
/apps/leuven/rocky8/icelake/2021a/
softwar
e/intel-compilers/2021.2.0/compile
r/2021.2.0/linux/bin/intel64/icc
cp: cannot stat '/apps/leuven/trai
ning/test': No such file ordirecto
ry
Hello World
```

Resource Summary

Stdout

Output File

Example

```
$ scontrol show job --clusters=wice 60049330
JobId=60049330 JobName=f70.slurm
   UserId=vsc3…(253…) GroupId=vsc3…(253…)
   Account=lp_my_project QOS=lp_my_project
   JobState=PENDING Reason=QOSGrpBillingMinutes
   SubmitTime=2023-02-16T00:24:58 EligibleTime=2023-02-16T00:24:58
   Partition=batch NodeList= NumNodes=4-4 NumCPUs=288 NumTasks=288 CPUs/Task=1
   TRES=cpu=288,mem=979200M,node=4,billing=733
   MinCPUsNode=72 MinMemoryCPU=3400M
   WorkDir=/vsc-hard-mounts/leuven-data/3../vsc3…/Odonate_SOM
   StdErr=/vsc-hard-mounts/leuven-data/3../vsc3…/Odonate_SOM/slurm-60049330.out
   StdIn=/dev/null
   StdOut=/vsc-hard-mounts/leuven-data/3../vsc3…/Odonate_SOM/slurm-60049330.out
```

Diagnosis

Why is my job in pending/hold state?
Check out the “Reason” on Slurm docs: https://slurm.schedmd.com/resource_limits.html

Other Partitions on wICE

GPU

```
#SBATCH --partition=gpu
#SBATCH --nodes=1
#SBATCH --ntasks=18
#SBATCH --gpus-per-node=1
```

Big Memory

```
#SBATCH --partition=bigmem
#SBATCH --nodes=1
#SBATCH --tasks-per-node=72
#SBATCH --mem-per-cpu=28000M
```

Interactive Partition

- Accessible via command line and Open OnDemand
- To quickly compile, test, debug your (parallel) application
- To pre-/post-process your data and make visualizations
- Short queue time
- 5 dedicated nodes on wICE:
  - 64 cores per node
  - 1 GPU (=7 GPU instances)
  - 512 GB memory
- Max resource per user: 8 cores, 1 GPU instance, 16 hour walltime
- Free of charge

Interactive Job

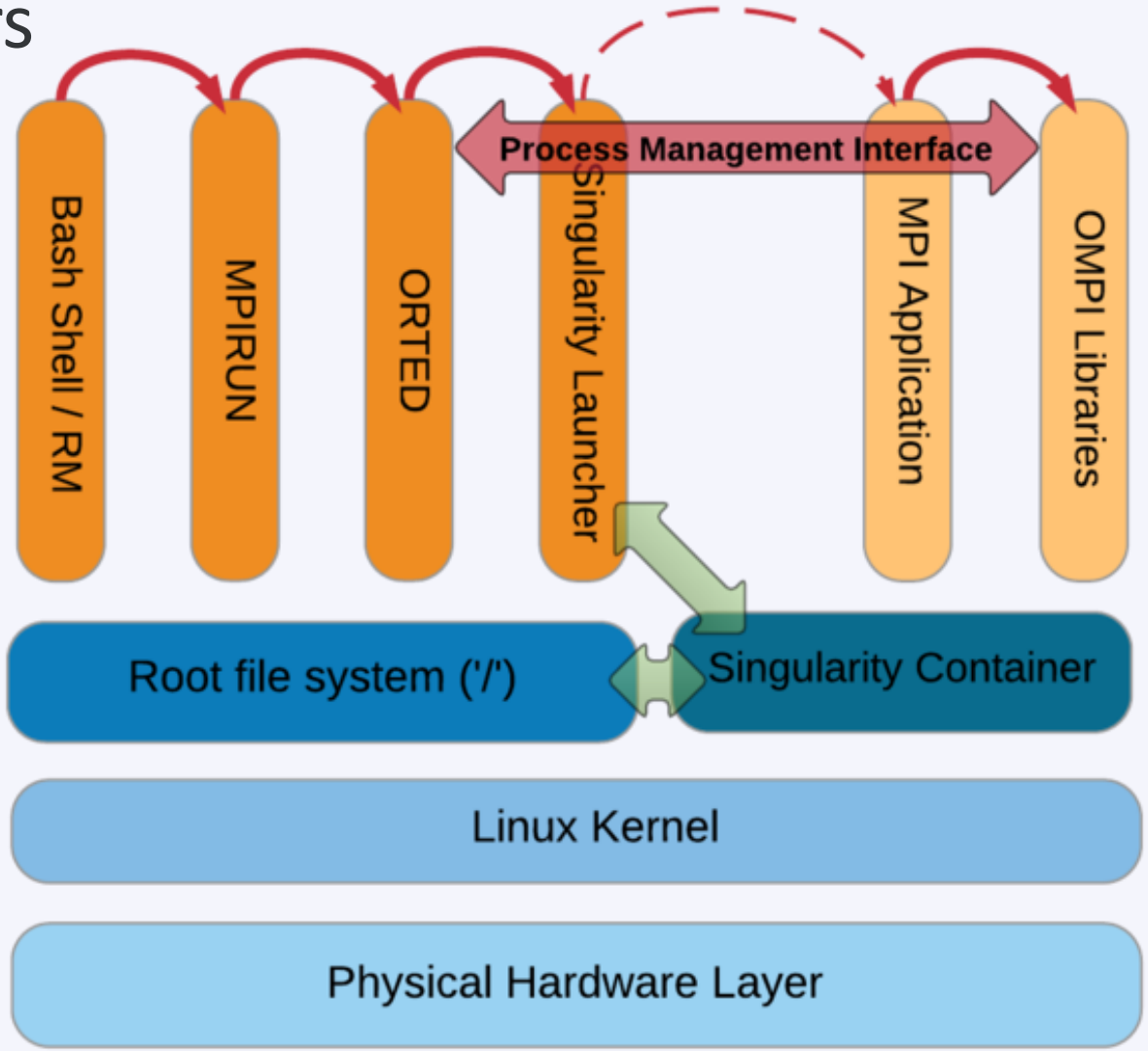
```
$ srun --account=lp_myproject --clusters=wice \
       --partition=interactive --nodes=1 --time=16:00:00 \
       --gpus-per-node=1  --pty bash -l
```

Apptainer (Singularity) Containers

- What?
  - Self-contained OS & software & data
- Why?
  - fully resolved dependency chains
  - portable workflow
- How?
  - You create the image
  - Run it on Genius/wICE
  - MPI/OpenMP is supported

script.slurm

```
#!/bin/bash -l
#SBATCH --time=30:00
#SBATCH --nodes=2
#SBATCH --ntasks=72
#SBATCH --clusters=wice
#SBATCH --account=lp_hpcinfo
apptainer run Project.simg
./model.exe
```

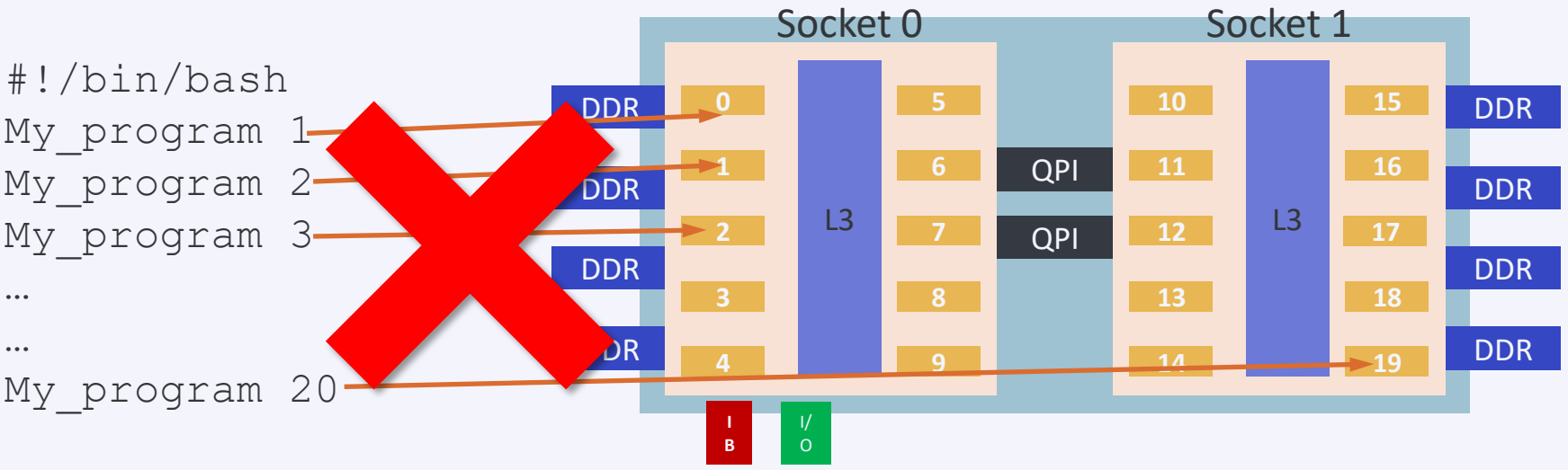


Worker Framework

Sequential…

- A shell program consists of a sequential list of commands
  -> bash script on HPC cluster will run sequentially

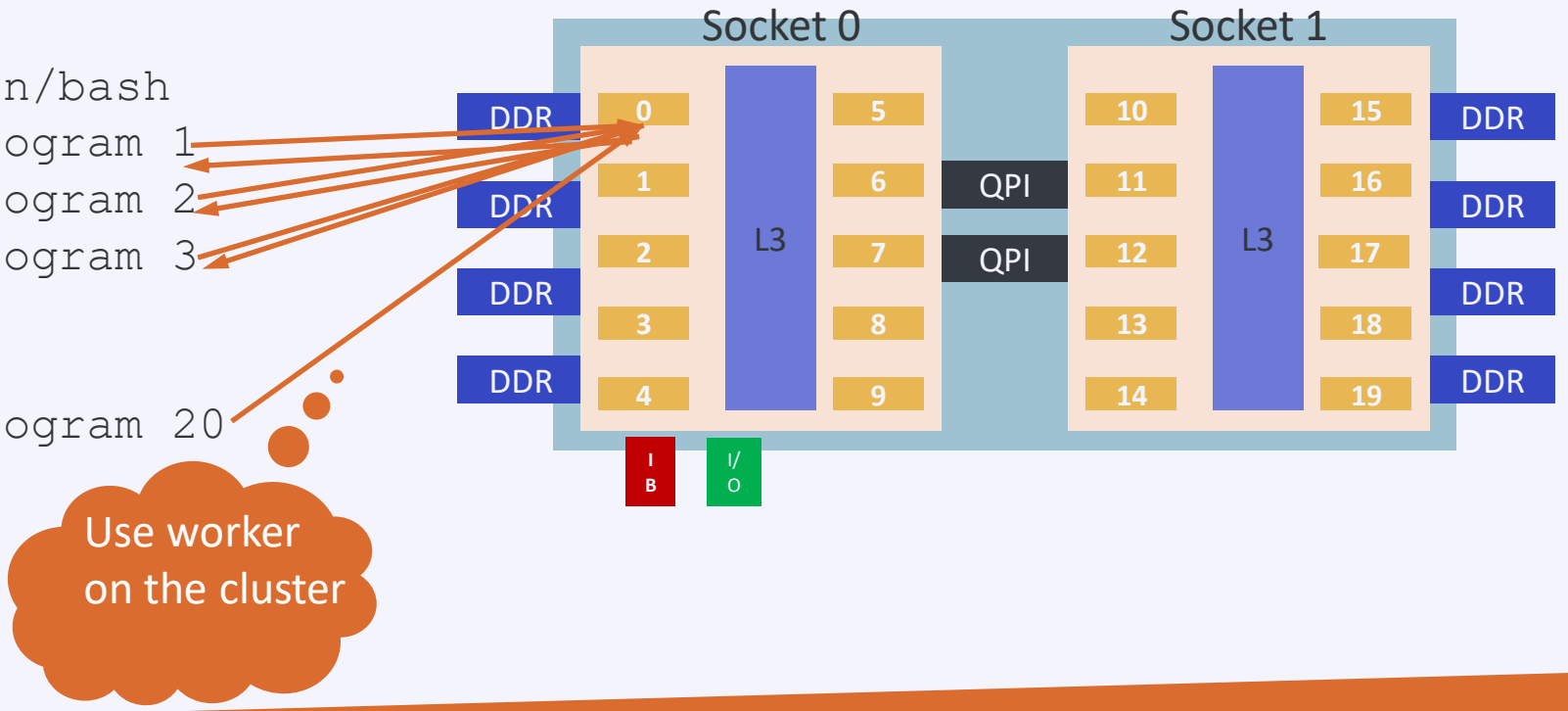
```
#!/bin/bash
My_program 1
My_program 2
My_program 3
…
…
My_program 20
```



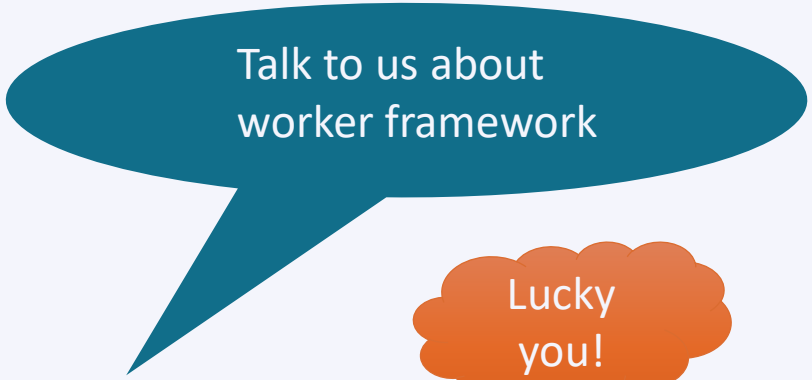
Sequential…

- A shell program consists of a sequential list of commands
  -> bash script on HPC cluster will run sequentially

```
#!/bin/bash
My_program 1
My_program 2
My_program 3
…
…
My_program 20
```



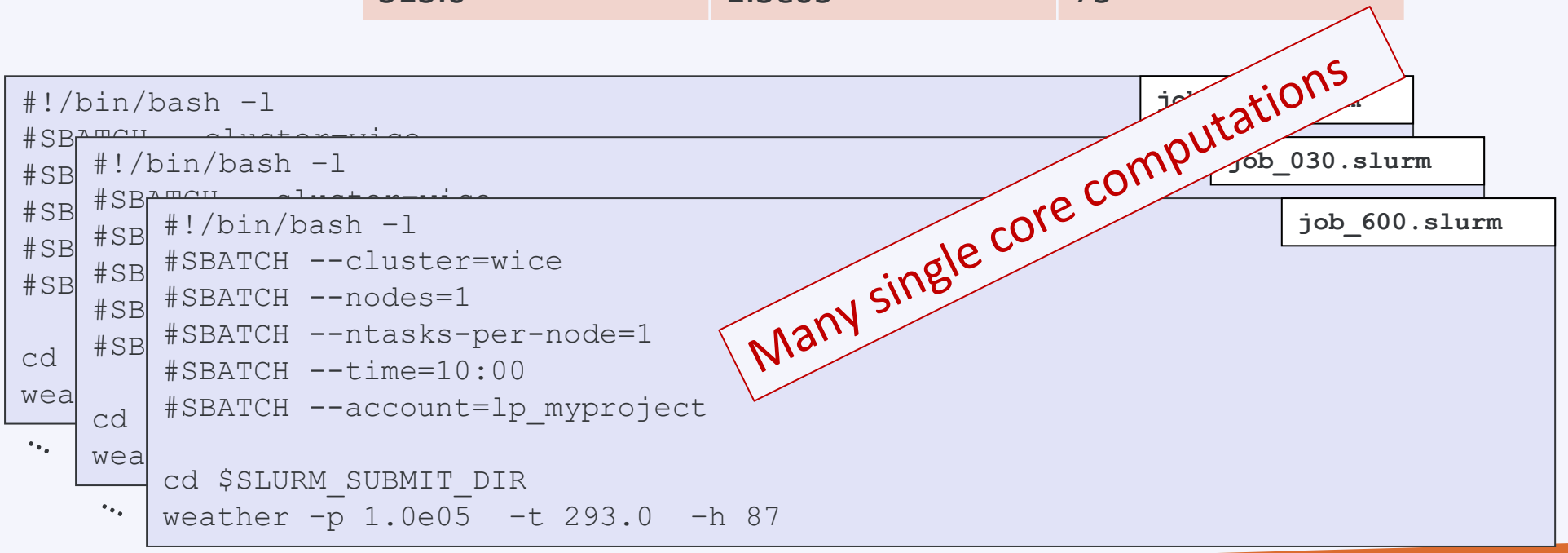
Parallel Computing



- Serial:
  - one program, on one core
- “Embarrassingly parallel” problems:
  - lots of runs of one program, with different parameters
- Problems that require “real” parallel algorithms
  - OpenMP
  - MPI : Message Passing Interface

Use case: parameter exploration

temperature	pressure	humidity
293.0	1.0e05	87
…	…	…
313.0	1.3e05	75



VLAAMS SUPERCOMPUTER CENTRUM

Solution: worker with `-data`

data.csv

temperature	pressure	humidity
293.0	1.0e05	87
…	…	…

job.slurm

```
#!/bin/bash -l
#SBATCH --cluster=wice
#SBATCH --nodes=2
#SBATCH --ntasks-per-node=72
#SBATCH --time=01:00:00
#SBATCH --account=lp_hpcinfo

cd $SLURM_SUBMIT_DIR
weather -p $pressure -t $temperature  -h $humidity
```

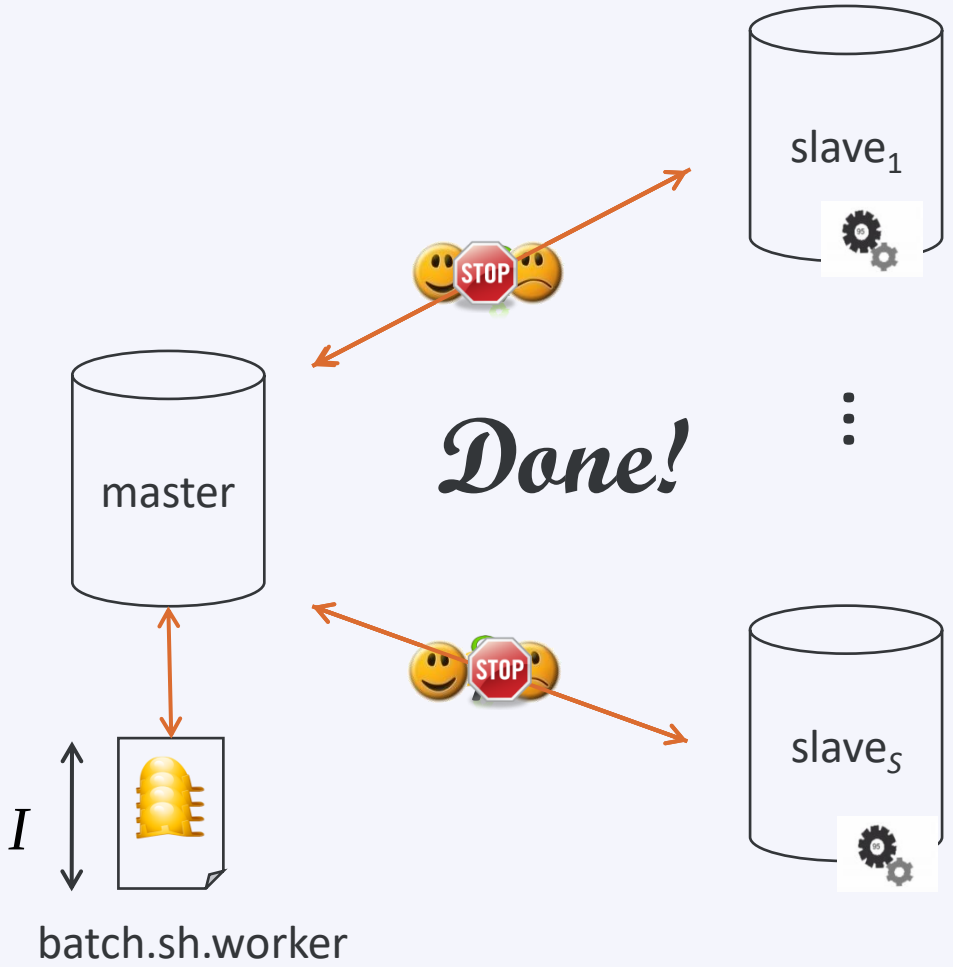
```
#From wICE interactive session:
$ module --force purge
$ module use /apps/leuven/rocky8/icelake/2021a/modules/all
$ module load worker/1.6.12-foss-2021a
$ wsub  -data data.csv  -batch job.slurm
```

Data exploration: steps

- Write slurm script with parameters
- Create Excel sheet with data
  - Convert to CSV format
- Submit with wsub
  - walltime is time to complete all work items

$$walltime_{job} \geq \frac{N \cdot walltime_{work\ item}}{nodes \cdot ntasks{-}per{-}node}$$

Worker processing: informally



slave queries for work

master sends work

slave notifies on success/failure, queries for more work

master sends stop

$$nodes = S + 1$$

How to use worker well?

- Many work items, i.e., #work items/#proc >> 1
- time(work item) > 1 minute
- Work item is not multithreaded
- Work item is multithreaded
  - will work, but user must be careful to request the right resources
  - Use –threaded <n> flag with wsub

- Here be dragons: licensing!

Worker & multithreading

- Some software uses multithreading automatically, e.g.,
  - R
  - Matlab
- Will use as many threads as there are cores, regardless of system load
  - 36 cores/node
  - 36 work items/node

} 72×72 threads/node

Oversubscription: *very* inefficient!!!

Controlling number of threads

- R, most of the time: `OMP_NUM_THREADS=1`

- Matlab
  - Use `maxNumCompThreads(1)` function call
  - Use compiler flag: `mcc -singleCompThread` …

program_pe.slurm

```
#!/bin/bash -l
#SBATCH -account=lp_myproject
#SBATCH --cluster=wice
#SBATCH --time=1:00:00
#SBATCH --nodes=2
#SBATCH --ntasks-per-node=72

module load R
cd $SLURM_SUBMIT_DIR

export OMP_NUM_THREADS=1
Rscript  program  $a  $b
```

Help on worker

- See documentation
  http://worker.readthedocs.io/
- Each command has help, use `-h`

- All resources have to be specified inside slurm script
- `wresume` does not work yet. Only `wsub` is so far supported
- `-master` flag does not work yet

Submitting worker jobs

From wICE (Interactive) Node

```
$ module av worker
---------------- /apps/leuven/rocky8/icelake/2021a/modules/all ----------
   worker/1.6.12-foss-2021a
```

From Genius Login Node

```
$ module av worker
---------------- /apps/leuven/rocky8/skylake/2021a/modules/all ----------

    worker/1.6.12-foss-2021a-wice    worker/1.6.12-intel-2021a  (D)
    worker/1.6.12-foss-2021a
```

Submitting worker jobs

- Let us use one of the worker examples:
  https://github.com/gjbex/worker/tree/development_slurm/examples/bash_example

alp.slurm

```
#!/bin/bash -l
#SBATCH --nodes=1
#SBATCH --ntasks-per-node=27
#SBATCH --time=00:30:00
#SBATCH --account=lpt2_sysadmin
#SBATCH --clusters=wice
cd $SLURM_SUBMIT_DIR
alphabet.sh $letter $counter
```

alpha.csv

```
letter,counter
a,1
b,2
c,3
d,4
e,5
f,6
....
```

Submitting worker jobs

- From Genius

```
#!/bin/bash -l
module purge
module use /apps/leuven/rocky8/skylake/2021a/modules/all
module load worker/1.6.12-foss-2021a-wice
wsub -batch alp.slurm -data alpha.csv
```

```
total number of work items: 26
Submitted batch job 60008899 on cluster wice
```

Submitting worker jobs

- From interactive job on wICE

```
#!/bin/bash -l
module --force purge
module use /apps/leuven/rocky8/icelake/2021a/modules/all
module load worker/1.6.12-foss-2021a
wsub -batch alp.slurm -data alpha.csv
```

```
total number of work items: 26
Submitted batch job 60008889 on cluster wice
```

Demo: Test yourself

Demo/test yourself

- Request membership to lp_hpcinfo group (account.vscentrum.be)
- File transfer with Filezilla
- Login with NX
- Check disk quota
- Check the credits
- Check/load/list/unload/purge module

Demo/test yourself

- Copy training material `/apps/leuven/training/HPC_intro/` to your `$VSC_DATA`
- Submit `cpujob` to the cluster
- List all your jobs `squeue -M wice`
- Check the information about the cpujob `slurm_jobinfo -M wice <job_ID>`
- Modify the `mpi.slurm` script to request 1 node, 72 cores for 30 minutes and get the notification about job start/end by e-mail
- Check the status of all the jobs

Demo - monitoring

- Submit an interactive job
  - Run your program on a compute node
  - Open a new terminal and ssh to a compute node
  - Check the resources usage (`htop`)
- Submit a batch GPU job
  - While the job is running get the information about the node `slurm_jobinfo...`
  - and check usage of resources on the node `ssh`, `top`, `nvidia-smi`

Demo - worker

- Copy intro training files (`/apps/leuven/training/worker/`) to your `$VSC_DATA`
- Go to `exercise1` directory
- Submit worker job
- Check the output file

Questions

Helpdesk:
hpcinfo@kuleuven.be or https://admin.kuleuven.be/icts/HPCinfo_form/HPC-info-formulier

VSC web site:
http://www.vscentrum.be/

VSC documentation: https://docs.vscentrum.be/en/latest/

VSC agenda, training sessions, events: https://www.vscentrum.be/vsctraining

Systems status page:
http://status.vscentrum.be

Stay Connected to VSC

LinkedIn®