



Vlaanderen
is supercomputing

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

By registering and participating in VSC trainings, you conform to the [VSC trainings code of conduct.](#)



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

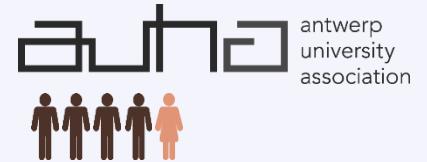
- Using Open OnDemand
- Software
- Miniconda
- Batch Jobs
- Worker
- Demo: test yourself



VSC

(Vlaams Supercomputer Centrum)

VSC PARTNERSHIP



Supported by



Surf to: www.vscentrum.be

Very rich suite of
courses every
academic year

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



[Home](#) [About VSC](#) [Systems & Services](#) [Showcase](#) [News & Events](#) [VSC Training](#) [User Portal](#) [Access](#)

Search... 

Welcome to VSC Flanders' most highly integrated high-performance research computing environment

[Read More](#)



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.vscenrtum.be>

Welcome to the VSC documentation

The VSC documentation offers extensive *how-to* guides and technical information about the services provided by the [Vlaams Supercomputer Centrum](#).

Accounts and access
How to get your VSC account and access the different VSC services and platforms.

Research Data
Data transfer and storage in the VSC infrastructure.

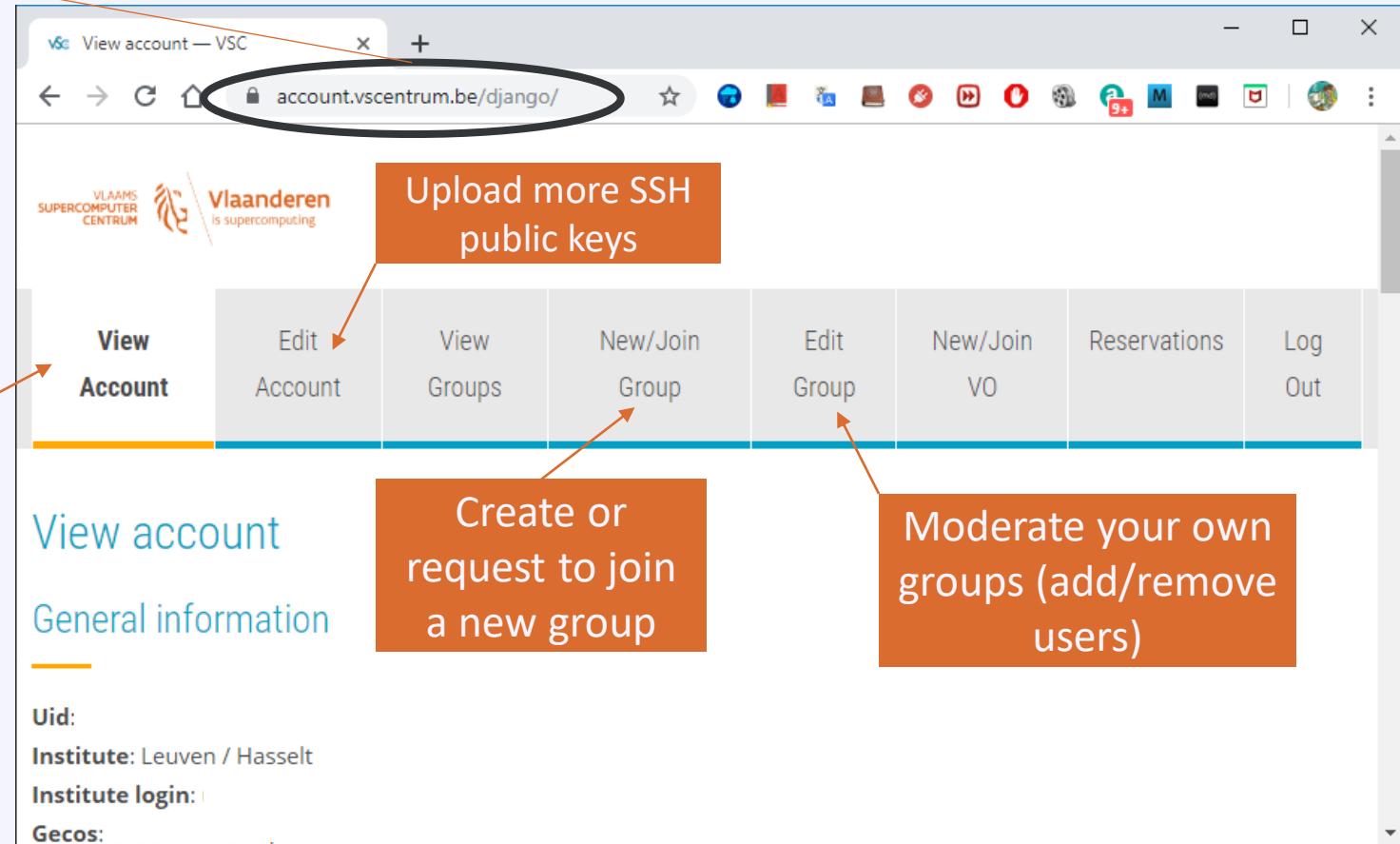
Compute
The high-performance computing (HPC) platform provides multiple tiers of parallel processing

Tier-1 Cloud
The VSC Cloud component provides *on-demand* resources in a more flexible and cloud-like

Tier-1 Data
The VSC Data component enables research data to remain close to the computing infrastructure

Search your keywords here; e.g. SSH key

To manage your
VSC account:
account.vscentrum.be



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

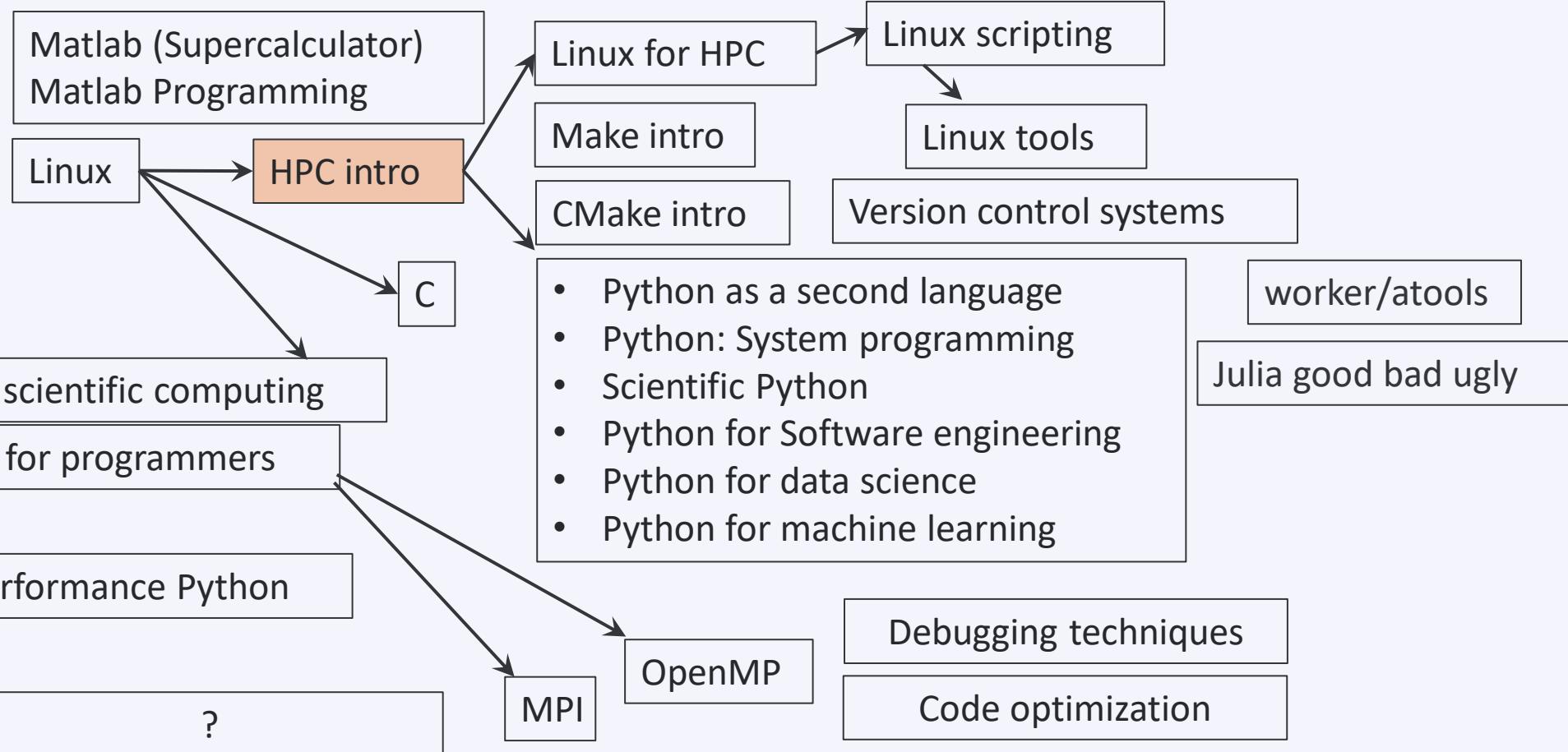
- (Optional) Create a secure (4096 bit) [SSH key pairs](#)
Upload it on the account page: www.account.vscentrum.be
- You need to [request a VSC account](#)
Normally processed swiftly
- Request [introductory credits](#) (2M free credits for 6 months)
- Request [project credits](#) (for supervisors and 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)

The screenshot shows a web browser window titled 'View account — VSC'. The URL is 'account.vscentrum.be/django...'. The page header includes the Vlaams Supercomputer Centrum logo and the text 'Vlaanderen is supercomputing'. A navigation menu at the top has tabs for 'View Account' (which is selected), 'Edit Account', 'View Groups', 'New/Join Group', 'Edit Group', 'New/Join VO', 'Reservations', and 'Log Out'. Below the menu, the main content area is titled 'View account' and 'General information'.

The screenshot shows a web browser window titled 'ICTS Servicecatalogus'. The URL is 'icts.kuleuven.be/sc'. The page header includes the KU Leuven logo and the text 'ICTS SERVICECATALOGUS'. A navigation menu has options for 'Home', 'ICTS SERVICECATALOGUS', 'WAT IS DE ICTS SERVICECATALOGUS?', 'OPGELET', and 'ZOEK IN DE SERVICECATALOGUS'. The 'OPGELET' section contains text about netto prijzen and overhead. On the right side, there is an image of a computer mouse.

VSC training

- Introductory



- Intermediate

- Advanced

- Specialized track

Infosessions:

- Containers
- Notebooks

worker/atools

Julia good bad ugly

- Python as a second language
- Python: System programming
- Scientific Python
- Python for Software engineering
- Python for data science
- Python for machine learning

Debugging techniques

Code optimization

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-2 Clusters @ KU Leuven

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

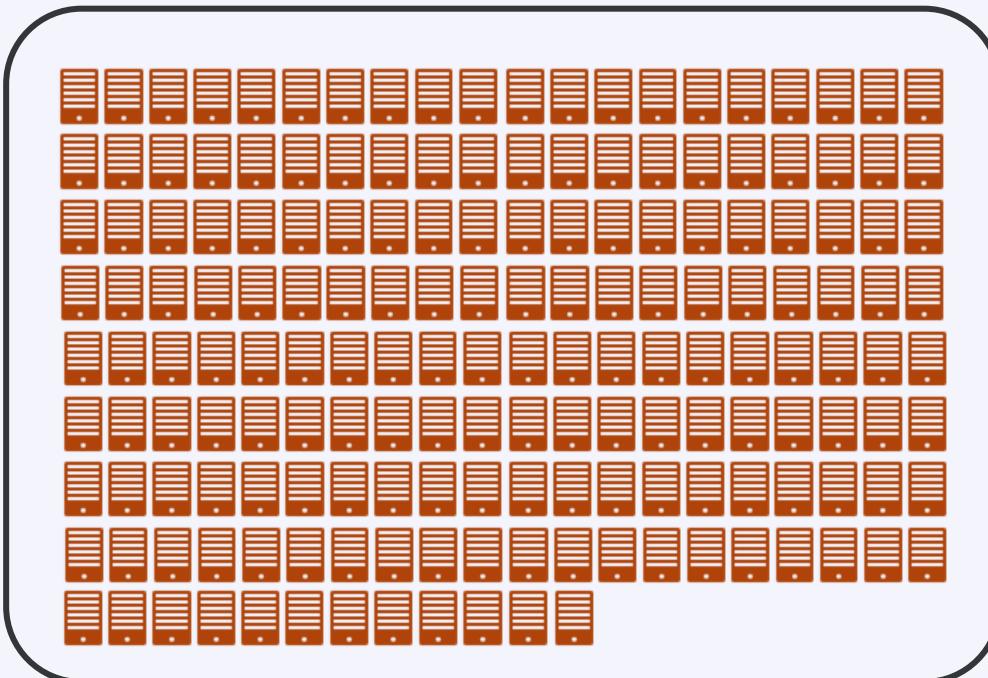


wICE (since 9/2022)
186 nodes; 13,392 cores



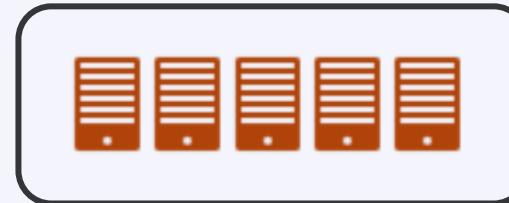
Tier-2 Cluster - wICE

Compute nodes



172x IceLake 72c 256 GB

Large memory nodes



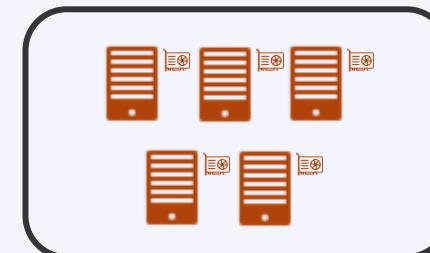
5x IceLake 72c 2048 GB

GPU nodes



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

Interactive nodes



5x IceLake 64c 512 GB
1 A100 80GB

No Dedicated
Login Nodes

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

Example

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.

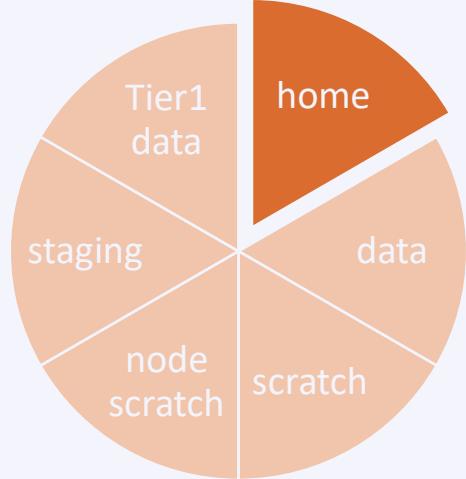
```
$ cd $VSC_DATA
```

Always check your storage balance using myquota command

```
$ 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
```

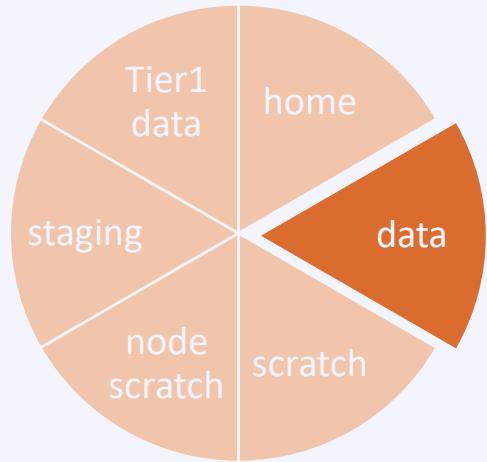
- [Request form for extra storage](#)
- [More information](#)

Storage



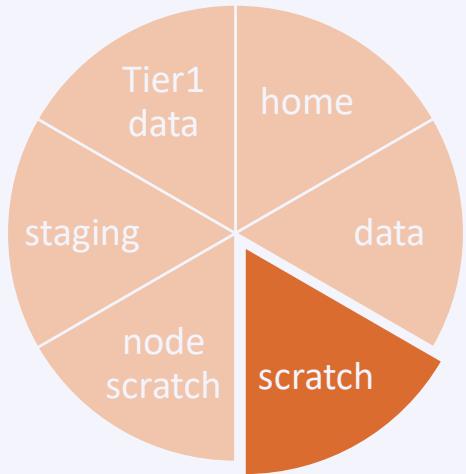
| Storage | home folder |
|-----------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Env. Variable | \$VSC_HOME |
| Filesystem Type | NFS |
| 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 | <ul style="list-style-type: none">- Stay away from using it- Can easily overflow:<ul style="list-style-type: none">+ Your jobs may crash+ Login issues |

Storage



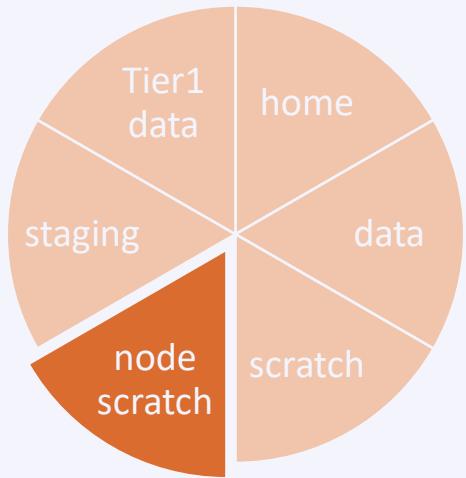
| Storage | data folder |
|-----------------|---------------------------------------------------------------------------------------------------------------------------------------------------|
| Env. Variable | \$VSC_DATA |
| Filesystem Type | NFS |
| Access | Global |
| Backup | Hourly, daily, weekly (until last month) inside the .snapshot folder. |
| Default Quota | 75 GB |
| Extension | On purchase |
| Usage | Your data, code, software, results |
| Remarks | <ul style="list-style-type: none">- Permanent storage for initial/final results- Not optimal for intensive or parallel I/O |

Storage



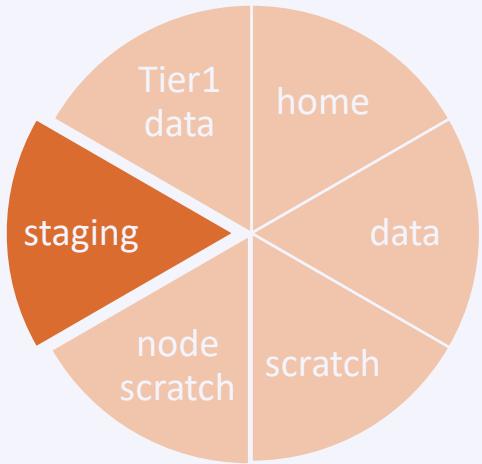
| Storage | scratch folder |
|-----------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Env. Variable | \$VSC_SCRATCH |
| Filesystem Type | Lustre |
| Access | Local |
| Backup | delete after 30 days from last access |
| Default Quota | 500 GB |
| Extension | Free |
| Usage | Intensive, parallel I/O, temporary files |
| Remarks | <ul style="list-style-type: none">- Recommended storage for all jobs- Copy scratch files to VSC_DATA or local storage after jobs are done- Deleted files cannot be recovered |

Storage



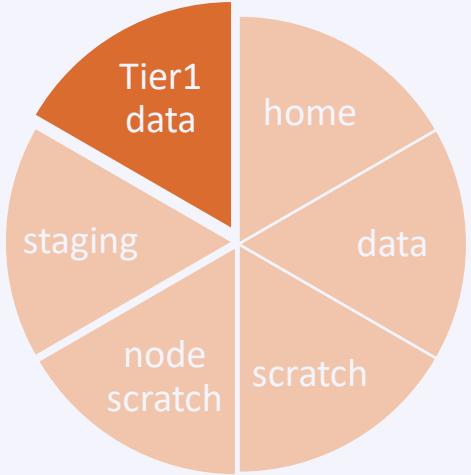
| Storage | Node scratch folder |
|-----------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Env. Variable | \$VSC_SCRATCH_NODE |
| Filesystem 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 | <ul style="list-style-type: none">- 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 | <ul style="list-style-type: none">- 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

Tier-2 credit system - what

- What is a credit
 - Credits are a measure for compute time

| Cluster / Partition | Credits/h |
|------------------------------------|-----------|
| Genius Skylake / Cascadelake / AMD | 10 000 |
| Genius Skylake BigMem | 12 000 |
| Genius Skylake 4 GPUs | 20 000 |
| Genius Cascadelake 8 GPUs | 39 900 |
| Genius Superdome | 126 000 |
| wICE Icelake thin node | 11 000 |
| wICE Icelake Large Memory | 19 000 |
| wICE IceLake GPU node | 45 000 |
| wICE interactive node | / |

Check ICTS service catalogue :
<https://icts.kuleuven.be/sc/english/research/HPC>

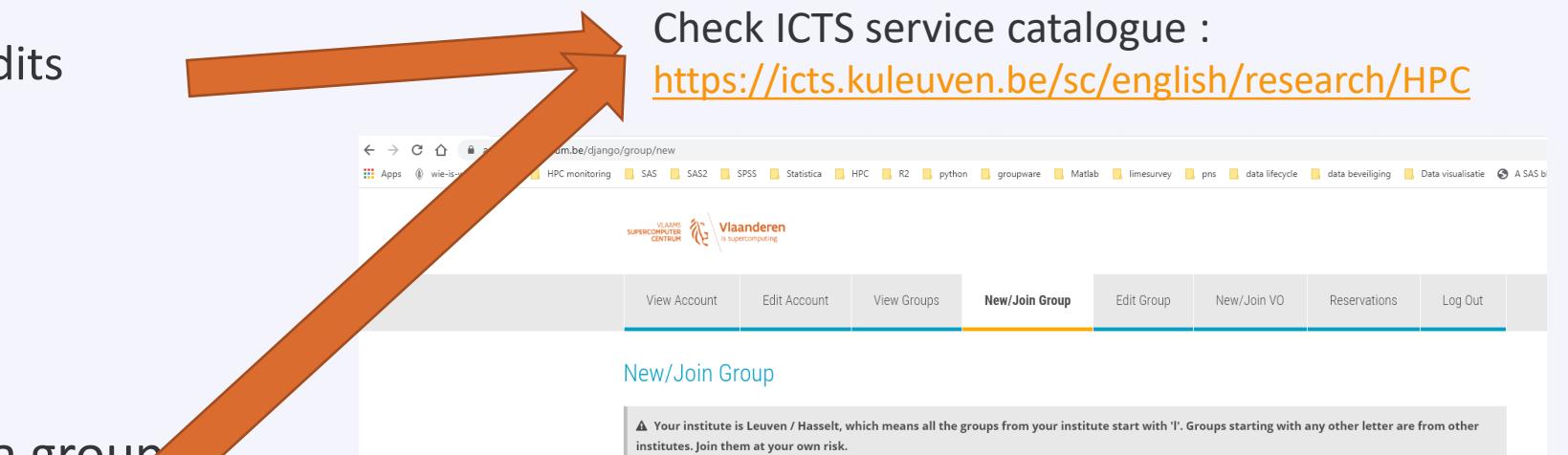
- Shared nodes are charged based on used fraction

Tier-2 credit system - why

- Why are credits used
 - Used to manage access the computational resource
 - Traceability, accountability
- Why are credits charged
 - Small fraction
 - Optimizing your code/workflow will result in spending less credits

Tier-2 credit system – how to get credits

- Request your introduction credits
 - Everybody
 - Free of charge
 - 1 time
 - 6 months



- Request permission to join a group

- Request project credits
 - Create group on the accountpage
 - Start the name preferably with p_
 - Your group will be lp_
 - Who creates the group can add or remove users
- Tier-2 credits are shared over all Tier-2 system

A screenshot of a web browser displaying the account page at <account.vscentrum.be/django/group/new>. The page has a header with 'View Account', 'Edit Account', 'View Groups', 'New/Join Group', 'Edit Group', and 'New/Join VO'. The 'New/Join Group' button is circled in orange. Below it, a 'Create new group' section contains a warning message: '⚠ We will automatically prepend the letter 'I' to your groupname.' and a 'Groupname *' input field. A large orange arrow points from the bottom left towards this 'Create new group' section.

Tier-2 credit system - how

- Submit your jobs with reference to the correct project
 - `#SBATCH -A lp_myproject`
 - `sbatch ... -A lp_myproject`
- Before submitting the job, the amount of available credits is checked with the maximum requested credits for the job
 - Maximum charge for the submitted jobs is calculated
 - $Credits = \text{minutes} * \text{floor}(\#\text{cpu cores} * \text{billing_weight_CPU} + \#\text{gres} * \text{billing_weight_gres})$
 - Taking into account any running jobs
 - When the job finishes, the actual cost is charged

Tier-2 credit system – manage your credits

- Slurm Accounting Manager => sam- commands

| Command | Purpose |
|--------------------------|--------------------------------------------|
| sam-balance [-A account] | List active projects and available credits |

```
tier2-p-login-3$ sam-balance -A lp_hpcinfo
ID      Name          Balance      Reserved      Available
=====  ======  ======  ======  ======
96141   lp_hpcinfo    9782636        0       9782636
tier2-p-login-3$ [ ]
```

Tier-2 credit system – manage your credits

- Slurm Accounting Manager => sam- commands

| Command | Purpose |
|----------------------------------------|-----------------------------------------------------------|
| sam-list-allocations -A <account-name> | List the validity dates of different projects/allocations |


```
tier2-p-login-3$ sam-list-allocations -A lp_hpcinfo
AllocID AccountID Account           Timestamp          Credits
===== ====== ====== ====== ======
80745    96141    lp_hpcinfo   2023-02-14T10:40:50  2000000
203565   96141    lp_hpcinfo   2023-04-17T16:42:26  2000000
227730   96141    lp_hpcinfo   2023-05-03T09:10:29  2000000
313474   96141    lp_hpcinfo   2023-06-19T17:33:59  8410000
tier2-p-login-3$ 
```

Tier-2 credit system – manage your credits

- Slurm Accounting Manager => sam- commands

| Command | Purpose |
|------------------------------------------------------------------------|--------------------------------------------|
| sam-quote sbatch [sbatch arguments] [my_job.slurm] | Shows the amount of credits a job will use |
| tier2-p-login-3\$ sam-quote sbatch --cluster=wice hello.slurm 10980 | |
| tier2-p-login-3\$ █ | |

Tier-2 credit system – manage your credits

- Slurm Accounting Manager => sam- commands

| Command | Purpose |
|------------------------------------------------------------------|----------------------------------------------------------------|
| sam-statement -A <account-name> -s <start-date> -e <end-date> | Overview of credits used on each job in a specific time window |

```
tier2-p-login-3$ sam-statement -A lp_hpcinfo -s 2023-09-01 -e 2023-10-25
#####
# Includes Account=lp_hpcinfo
# Generated on Mon Oct 30 11:01:29 2023.
# Reporting fund activity from 2023-09-01 to 2023-10-25
#
#####

Credits deposited in the given period: 0
Credits refunded in the given period: 0
Credits consumed in the given period: 0

-----
JobID      Cluster   Account        User      Partition    Credits
=====  =====  =====  =====  =====  =====
60739746   wice     lp_hpcinfo    vsc30446 batch    0
60740123   wice     lp_hpcinfo    vsc30446 batch    0
60740125   wice     lp_hpcinfo    vsc30446 batch    0
60746730   wice     lp_hpcinfo    vsc30446 interactive 0
60773547   wice     lp_hpcinfo    vsc30446 interactive 0
60774141   wice     lp_hpcinfo    vsc30446 batch    0
60774160   wice     lp_hpcinfo    vsc30446 batch    0
60783857   wice     lp_hpcinfo    vsc30446 interactive 0
tier2-p-login-3$
```

Tier-2 credit system – manage your credits

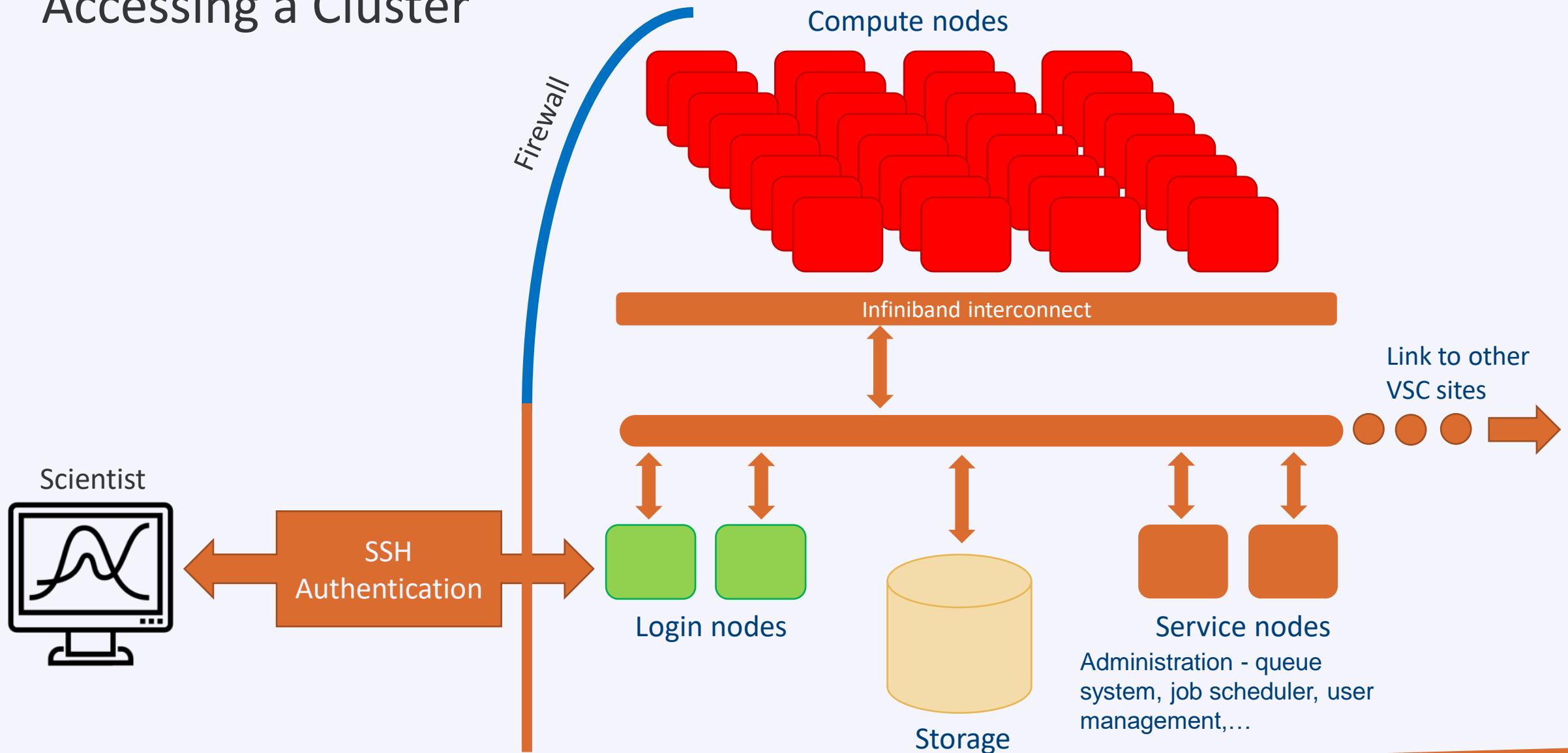
- Slurm Accounting Manager => sam- commands

| Command | Purpose |
|------------------------------------------------------------------|----------------------------------------------------------------|
| sam-balance | List active projects and available credits |
| sam-list-allocations -A <account-name> | List the validity dates of different projects/allocations |
| sam-quote sbatch [sbatch arguments] [my_job.slurm] | Shows the amount of credits a job will use |
| sam-statement -A <account-name> -s <start-date> -e <end-date> | Overview of credits used on each job in a specific time window |



Login Nodes & MFA

Accessing a Cluster



How to login?

| Open OnDemand | SSH client |
|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <ul style="list-style-type: none"><input type="checkbox"/> Web-based login via browser<input type="checkbox"/> https://ondemand.hpc.kuleuven.be<input type="checkbox"/> Multi-Factor Authentication (MFA): Login first to your institute/university<input type="checkbox"/> No SSH-key needed | <ul style="list-style-type: none"><input type="checkbox"/> Windows: PuTTY, MobaXterm, NoMachine MacOS/Linux: terminal, NoMachine<input type="checkbox"/> Requires (open)SSH keys<input type="checkbox"/> Upload your key to VSC account page wait at least 30 minutes<input type="checkbox"/> Hosts: login.hpc.kuleuven.be:22 nx.hpc.kuleuven.be (NX)<input type="checkbox"/> Use SSH agent (recommended) Use SSH Config (recommended)<input type="checkbox"/> Can open GUIs if X11 is configured |
| NX | |
| <ul style="list-style-type: none"><input type="checkbox"/> Graphical desktop<input type="checkbox"/> Using NoMachine client<input type="checkbox"/> Using GUI (Matlab, SAS, visualization)<input type="checkbox"/> Nvidia Quattro GPU for visualization | |

Using Login Nodes

- 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

Tips

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

Warning

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:

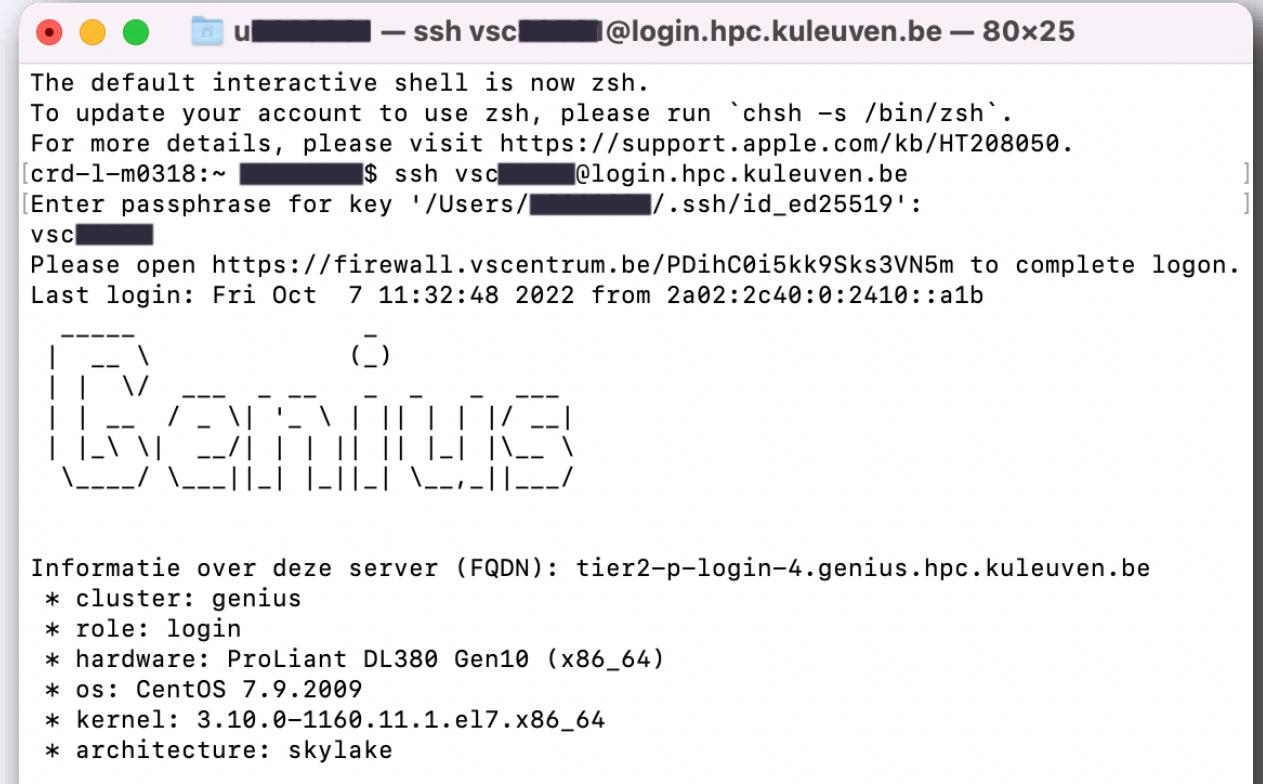
```
$ ssh vscXXXXX@<host_name>
```

- ✓ 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



The default interactive shell is now zsh.
To update your account to use zsh, please run `chsh -s /bin/zsh`.
For more details, please visit <https://support.apple.com/kb/HT208050>.
[crd-l-m0318:~ vscXXXXX\$ ssh vscXXXXX@login.hpc.kuleuven.be
[Enter passphrase for key '/Users/vscXXXXX/.ssh/id_ed25519':
vscXXXXX
Please open <https://firewall.vscentrum.be/PDihC0i5kk9Sks3VN5m> to complete logon.
Last login: Fri Oct 7 11:32:48 2022 from 2a02:2c40:0:2410::a1b
[REDACTED]
Informatie over deze server (FQDN): tier2-p-login-4.genius.hpc.kuleuven.be
* cluster: genius
* role: login
* hardware: ProLiant DL380 Gen10 (x86_64)
* os: CentOS 7.9.2009
* kernel: 3.10.0-1160.11.1.el7.x86_64
* architecture: skylake
✓ [okt/12 13:31] vscXXXXX@tier2-p-login-4 ~ \$

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:

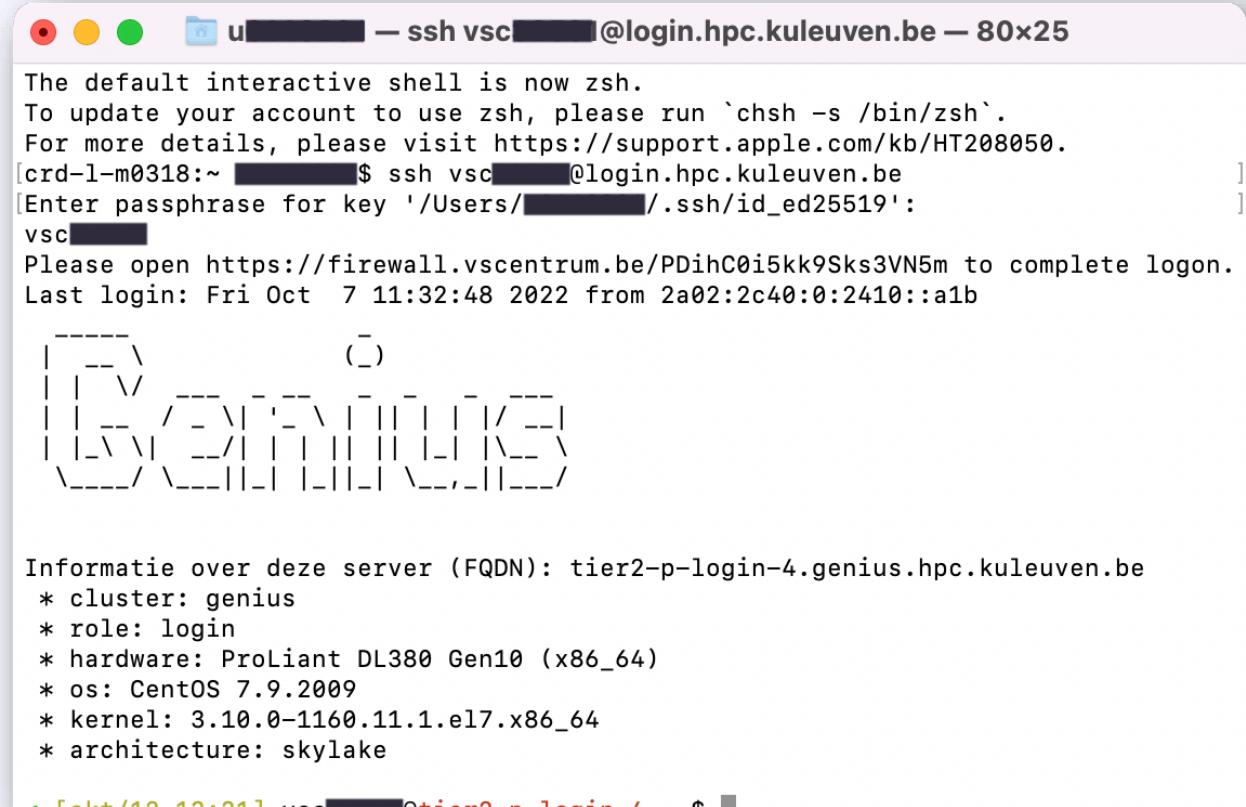
```
eval $(ssh-agent)
```

- ✓ If your key is not found,
add it to the Agent:

```
$ ssh-add </path/to/keyfile>
```

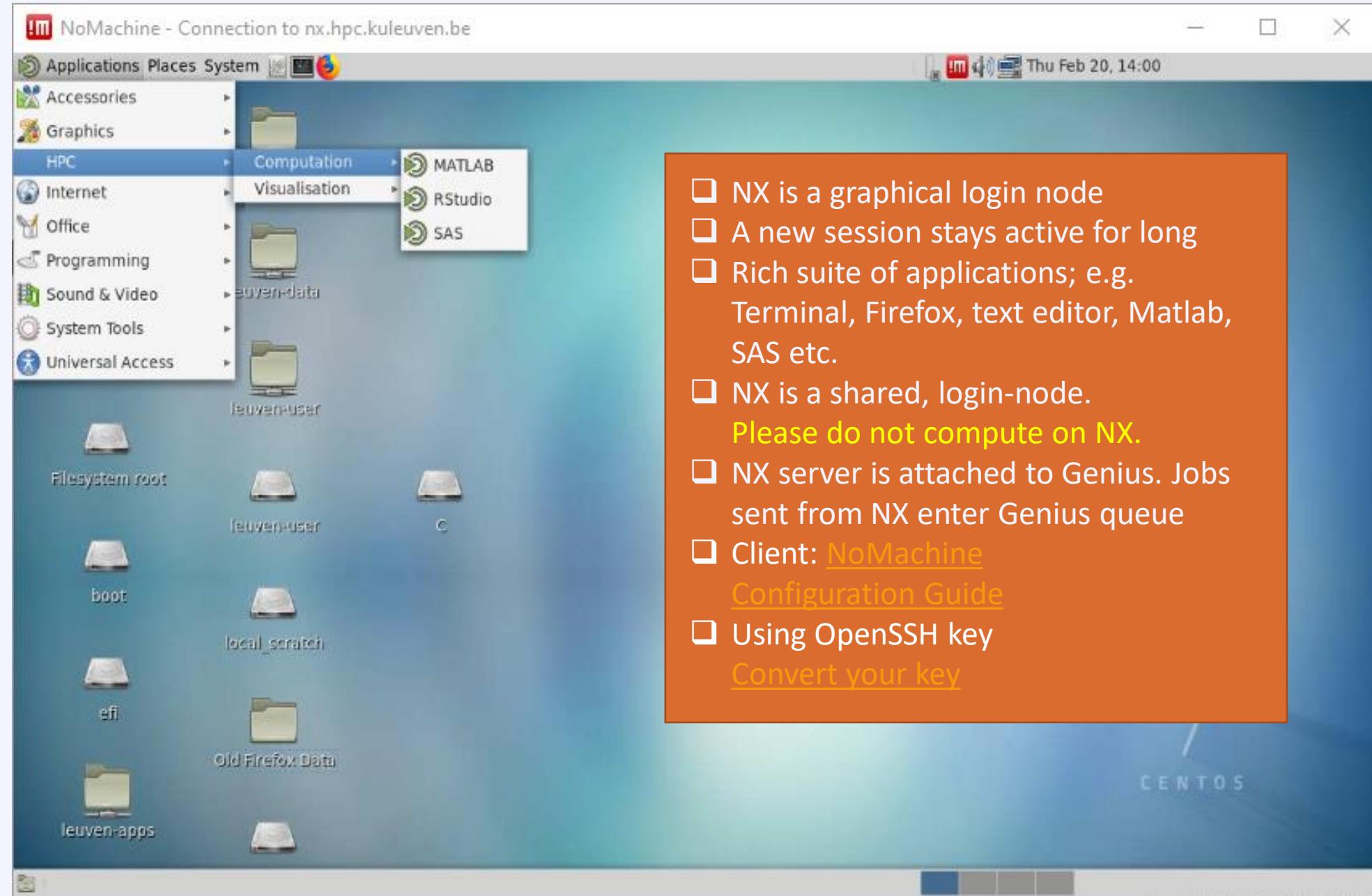
- ✓ Use ssh to connect:

```
$ ssh vscXXXXX@<hostname>
```



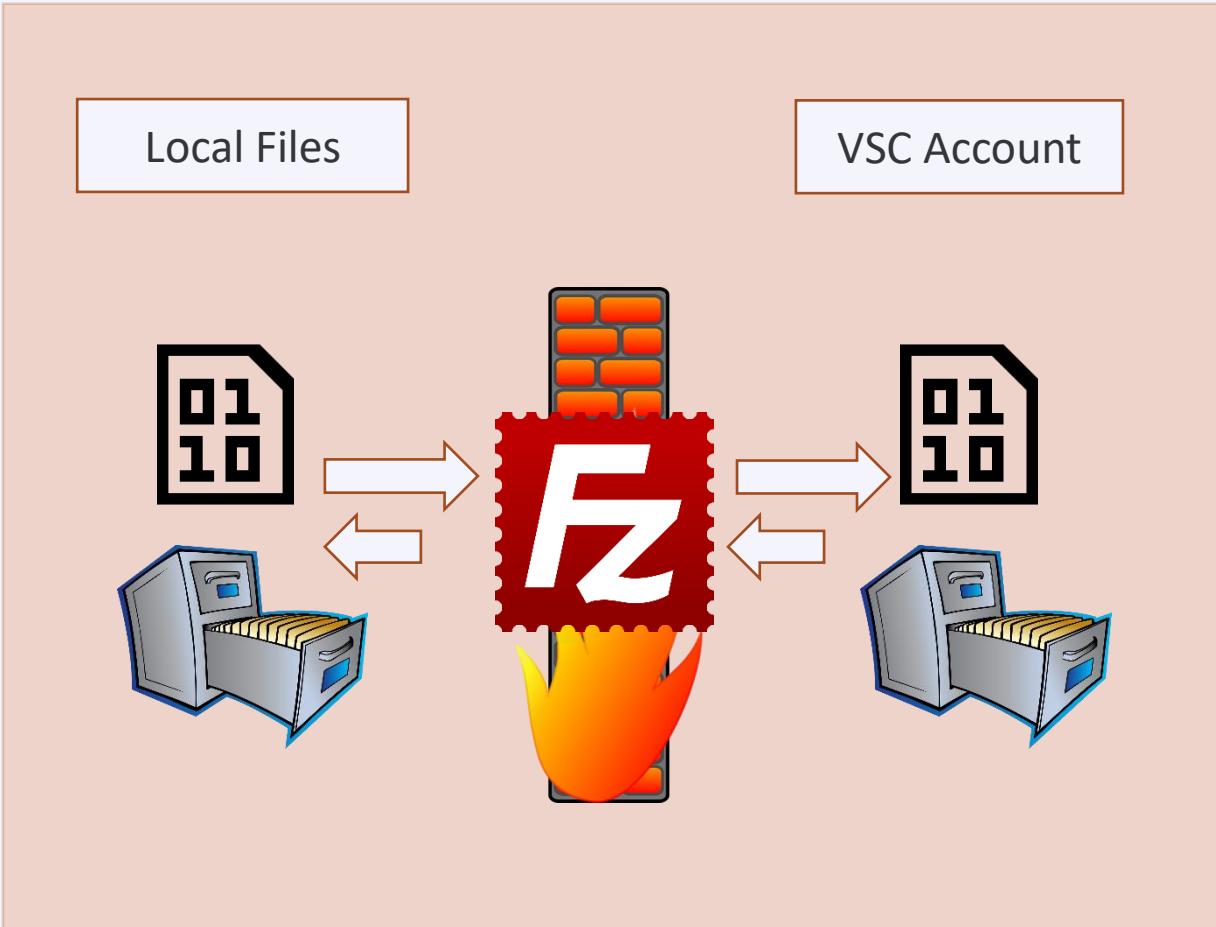
```
The default interactive shell is now zsh.  
To update your account to use zsh, please run `chsh -s /bin/zsh`.  
For more details, please visit https://support.apple.com/kb/HT208050.  
[crd-l-m0318:~] $ ssh vscXXXXX@login.hpc.kuleuven.be  
[Enter passphrase for key '/Users/[REDACTED]/.ssh/id_ed25519':  
vscXXXXX  
Please open https://firewall.vscentrum.be/PDihC0i5kk9Sks3VN5m to complete logon.  
Last login: Fri Oct 7 11:32:48 2022 from 2a02:2c40:0:2410::a1b  
[REDACTED]  
Informatie over deze server (FQDN): tier2-p-login-4.genius.hpc.kuleuven.be  
* cluster: genius  
* role: login  
* hardware: ProLiant DL380 Gen10 (x86_64)  
* os: CentOS 7.9.2009  
* kernel: 3.10.0-1160.11.1.el7.x86_64  
* architecture: skylake  
✓ [okt/12 13:31] vscXXXXX@tier2-p-login-4 ~ $
```

NX



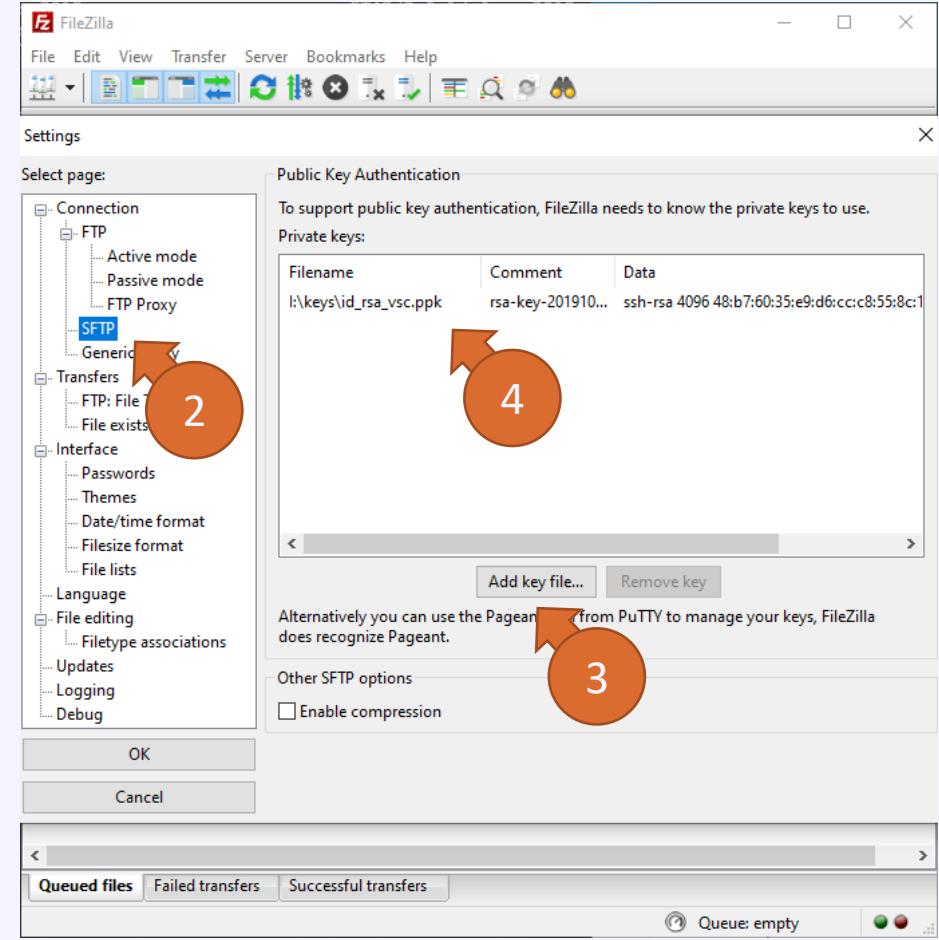
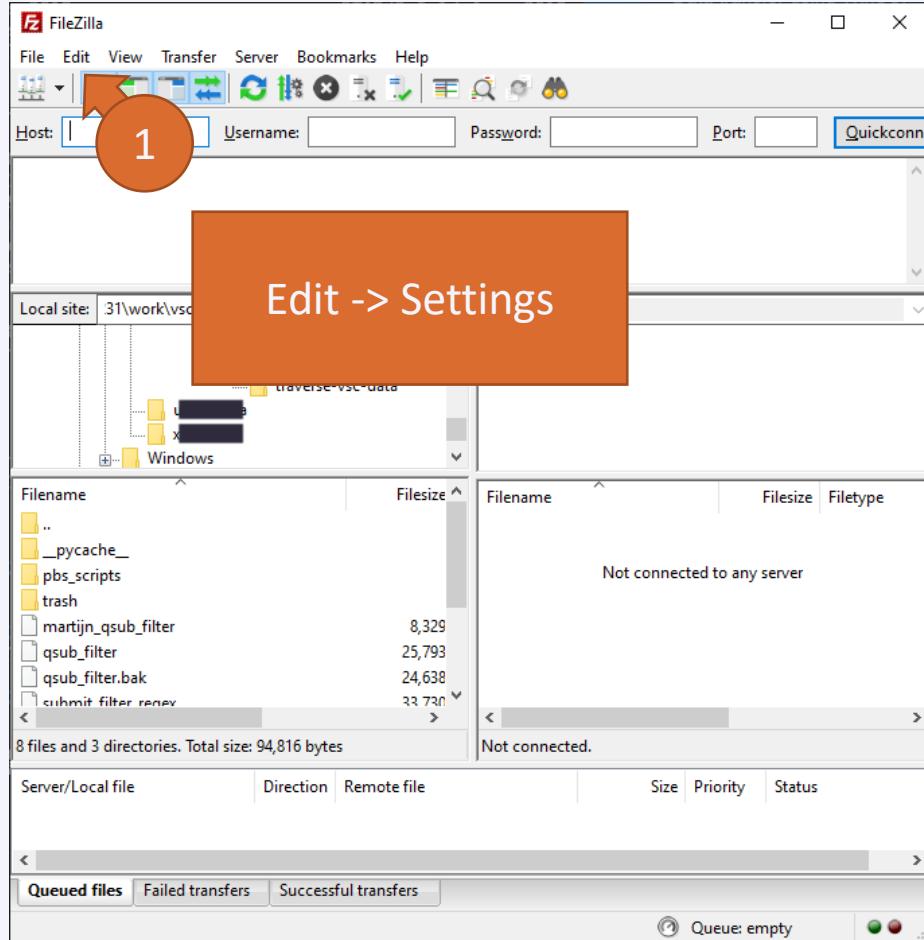
File Transfer with FileZilla

File transfer

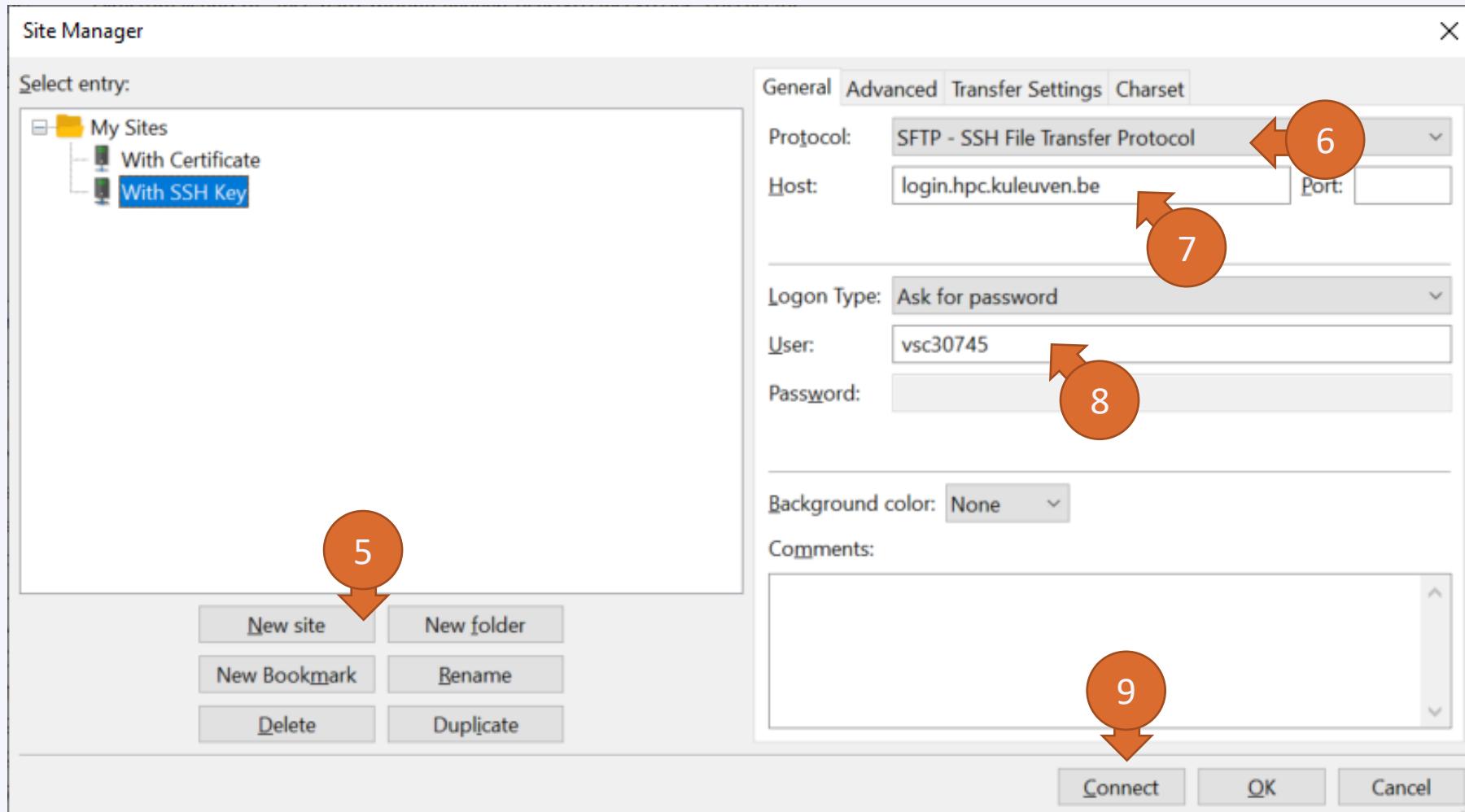


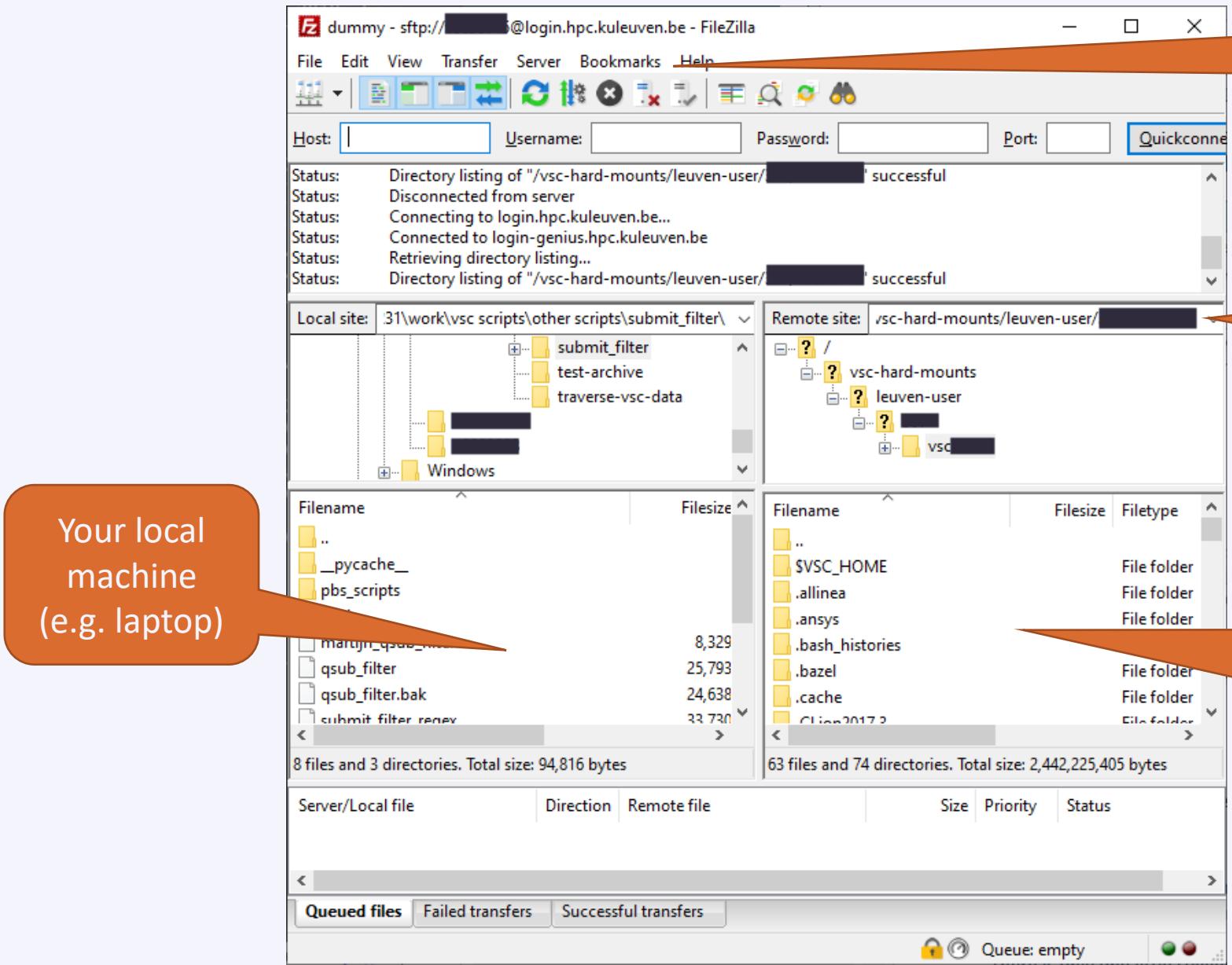
| Application | OS |
|---------------------------|---------------------|
| FileZilla | Windows, Linux, Mac |
| WinSCP | Windows |
| rsync, scp | Linux, Mac |
| Globus | Window, Linux, Mac |

File transfer



File transfer





Your local
machine
(e.g. laptop)

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

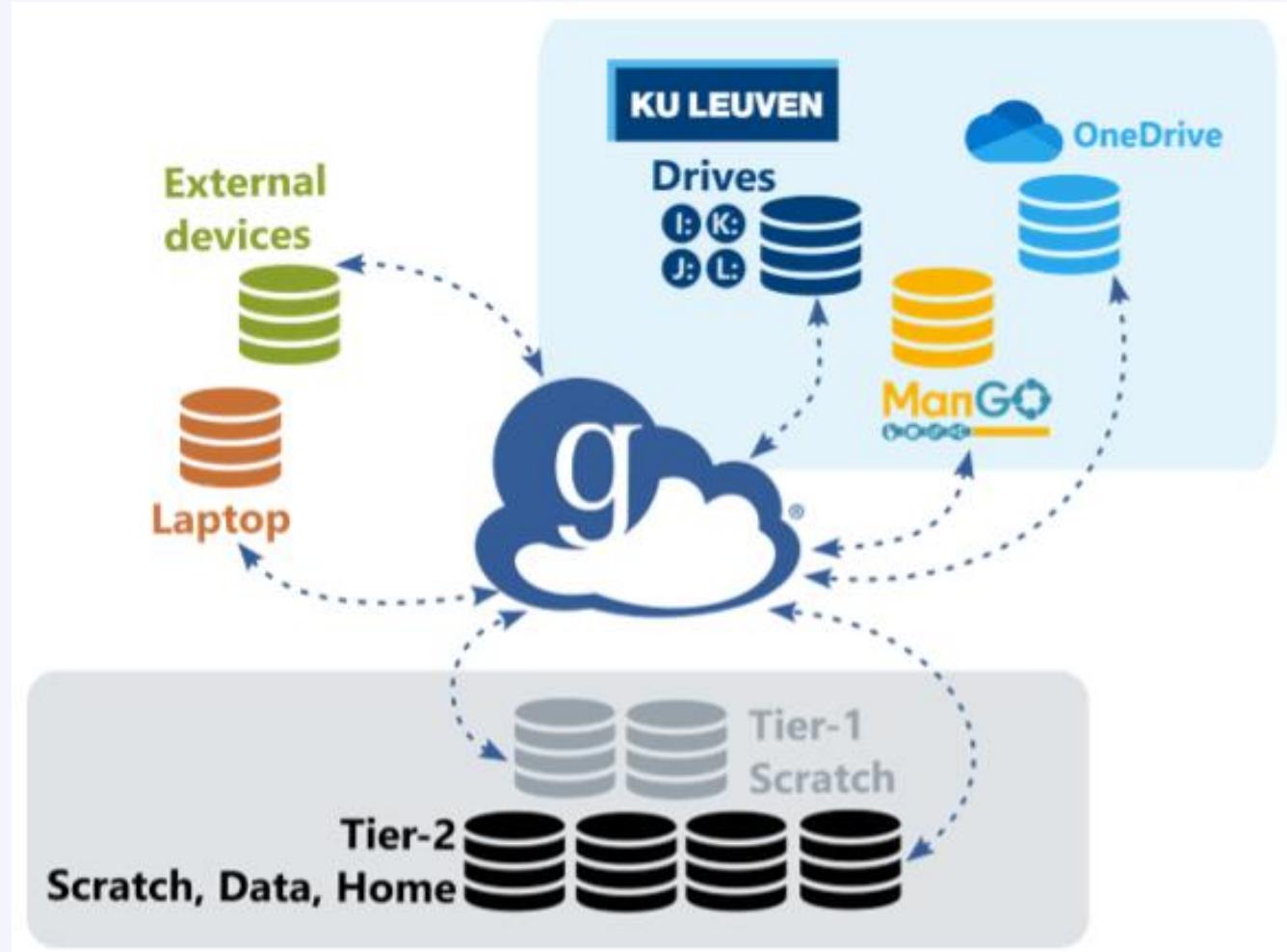
rsync

- ❑ 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
```

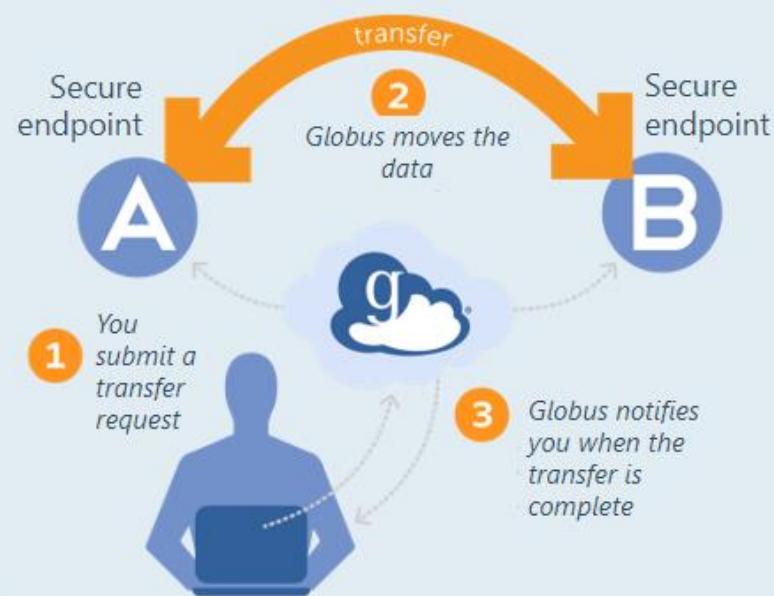
Globus

- 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

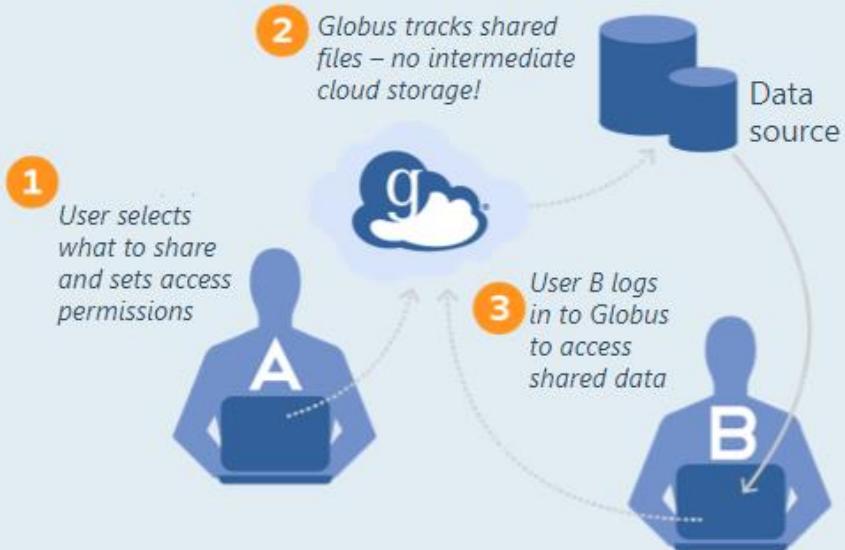


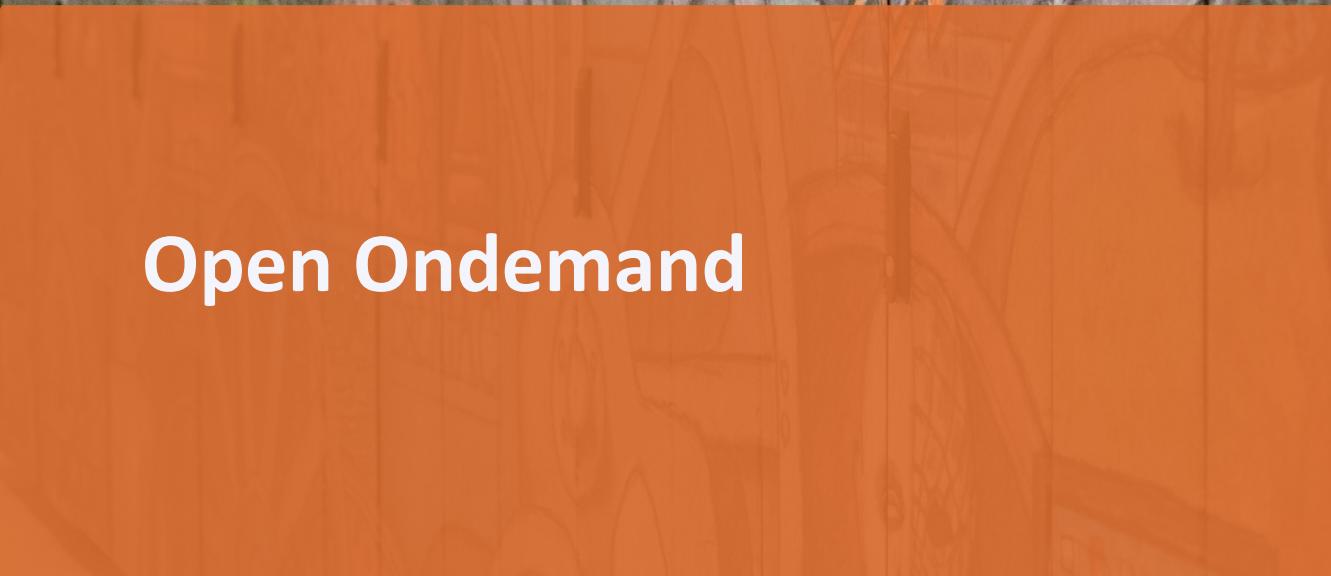


Transfer data



Share data





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

The screenshot shows the Kuleuven OnDemand web interface. At the top, there is a navigation bar with links for 'Kuleuven OnDemand', 'Apps', 'Files', 'Jobs', 'Clusters', 'Interactive Apps', and a help icon. Below the navigation bar, the page title is 'KU LEUVEN' and the subtitle is 'OnDemand provides an integrated, single access point for all of your HPC resources.' A section titled 'Pinned Apps' is described as 'A featured subset of all available apps'. The pinned apps are arranged in a grid:

| Icon | Name | Type |
|------------------------|---------------------------|------------------------------------|
| Clock icon | Active Jobs | System Installed App |
| House icon | Home Directory | System Installed App |
| Pencil icon | Job Composer | System Installed App |
| Greater than icon | Login Server Shell Access | System Installed App |
| VS Code icon | code-server | System Installed App |
| Interactive Shell icon | Interactive Shell | System Installed App |
| Jupyter Lab icon | Jupyter Lab | System Installed App |
| Rapids icon | RAPIDS | Nvidia Rapids System Installed App |
| R Studio icon | RStudio Server | System Installed App |
| Tensorboard icon | Tensorboard | System Installed App |



Open OnDemand

Access your VSC storage

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

The screenshot shows the KU Leuven OnDemand web interface. At the top, there is a navigation bar with tabs for "KU Leuven OnDemand", "Apps", "Files", "Jobs", "Clusters", "Interactive Apps", and a file icon. The "Files" tab is currently selected, and a dropdown menu is open. The menu contains two items: "Home Directory" (represented by a house icon) and "Data Directory" (represented by a folder icon), which is located at the path "/data/leuven/307/vsc30745".

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

KU Leuven OnDemand Apps ▾ Files ▾ **Jobs ▾** Clusters ▾ Interactive Apps ▾ ? ▾

① Active Jobs
Job Composer

Your Jobs ▾ All Clusters ▾

Active Jobs

Show 50 entries Filter:

| ID | Name | User | Account | Time Used | Queue | Status | Cluster | Actions |
|----|-------------------------|----------|------------|-----------|-------|-----------|---------|---------|
| | 55337242 PostProcessing | vsc30745 | lp_hpcinfo | 00:02:03 | batch | Completed | genius | |



Open OnDemand: Login Server Shell Access

- ✓ 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

Host: login.hpc.kuleuven.be Themes: Default

```
[REDACTED]
```

Informatie over deze server (FQDN): tier2-p-login-1.genius.hpc.kuleuven.be

- * cluster: genius
- * role: login
- * hardware: ProLiant DL380 Gen10 (x86_64)
- * os: Rocky 8.8
- * kernel: 4.18.0-477.10.1.el8_8.x86_64
- * architecture:
- * architecture-suffix:

On 25-09-2023 the operating system of the Genius cluster was upgraded to Rocky8.
The following documentation page provides an overview of important changes:
https://docs.vsczentrum.be/leuven/genius_2_rocky.html

Last login: Tue Nov 7 23:42:32 2023 from 2a02:2c40:0:2410::a1c

```
vsc30745@tier2-p-login-1 ~  
$ [REDACTED]
```



Open OnDemand: 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)

Interactive Shell

Interactive Apps

| Servers |
|-------------------|
| Interactive Shell |
| Jupyter Lab |
| Nvidia Rapids |
| RStudio Server |
| Tensorboard |
| code-server |

Cluster: wice

Account: lpt2_sysadmin

Partition: interactive

Number of hours: 1

Number of cores: 1

Required memory per core in megabytes: 3400

Number of nodes: 1

Number of gpu's: 0

[type:] Specify the total number of GPUs slices for the job. An optional GPU type specification can be supplied. For example "A100:3" or "3".

Reservation (optional):

Name of an existing reservation in which the job should run

I would like to receive an email when the session starts

Launch

* The Interactive Shell session data for this session can be accessed under the [data root directory](#).



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 (next slide)

Jupyter Lab

This app will launch a Jupyter Lab server on one or more nodes.

Interactive Apps

Servers

● Interactive Shell

● **Jupyter Lab**

■ Nvidia Rapids

■ RStudio Server

■ Tensorboard

■ code-server

Cluster

genius

Account

lpt2_sysadmin

Partition

gpu_p100

batch(_long) or bigmem or interactive or gpu(_p100|v100) or dedicated...

Enable nvidia rapids

Only available on w/ICE interactive or gpu nodes with a GPU requested

Number of hours

1

Number of cores

9

Required memory per core in megabytes

3400

Number of nodes

1

Number of gpu's

1

[type:] Specify the total number of GPUs slices for the job. An optional GPU type specification can be supplied. For example "A100:3" or "3".

Reservation (optional)

Name of an existing reservation in which the job should run

I would like to receive an email when the session starts

Pre-run scriptlet

Bash commands to be executed before starting application, \$node,_arch variable containing for example 'skylake' is available

Launch

* The Jupyter Lab session data for this session can be accessed under the [data root directory](#).



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
```

Permanently 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
```

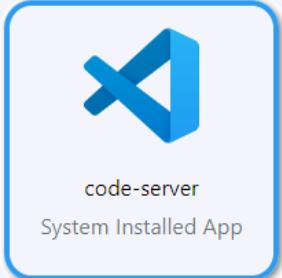
Step 3: Register Your Kernel

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

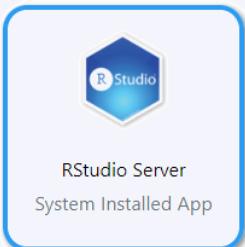
Start a new Jupyter Lab session, and the new kernels must be there



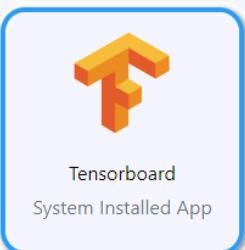
Open OnDemand: Other tools



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

Interactive Apps

| Servers |
|----------------------------------|
| ● Interactive Shell |
| Jupyter Lab |
| ● RStudio Server |
| ● Tensorboard |
| ● code-server |
| Work in progress |
| ● ParaViewWeb - Work in progress |
| ● cryosparc |

Jupyter Lab

This app will launch a Jupyter Lab server on one or more nodes.

Account

lp_myproject

Partition

interactive

batch(_long) or bigmem or interactive or gpu or dedicated...

Number of hours

3

Number of cores

1

Required memory per core in megabytes

3400

Number of nodes

1

Number of gpu's

0

[type:] Specify the total number of GPUs slices for the job. An optional GPU type specification can be supplied. For example "A100:3" or "3".

Reservation (optional)

Name of an existing reservation in which the job should run

I would like to receive an email when the session starts

Launch

Software Stack

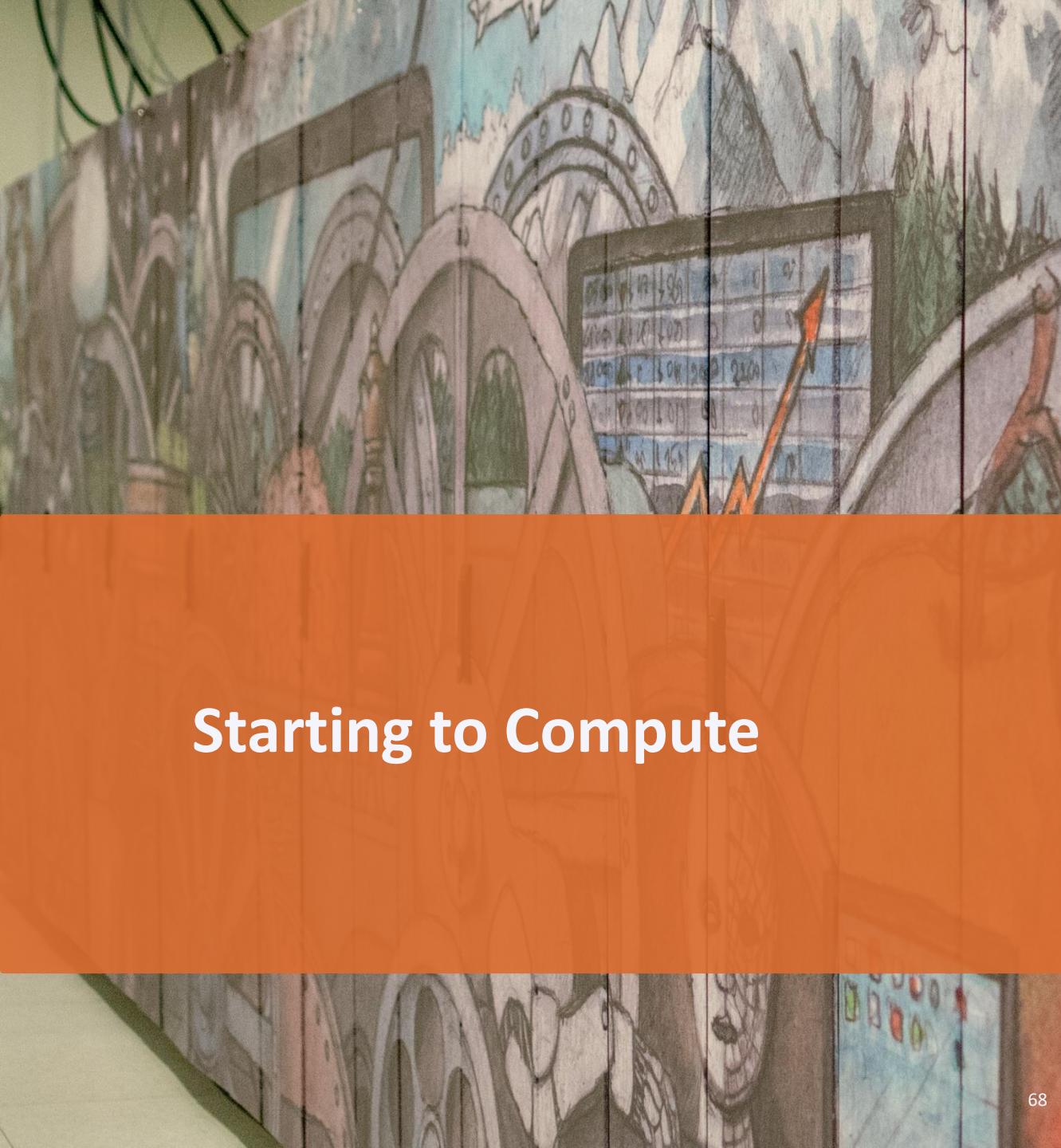
Software: Available Modules

- ✓ 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 wlCE 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

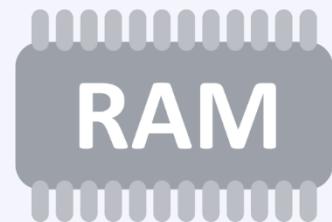
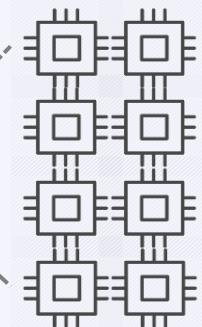
- ✓ You can always install your desired software in your \$VSC_DATA
Use Intel or FOSS toolchains
- ✓ Compile your code on a compute node (with interactive job)
- ✓ If you cannot, ask us for help
- ✓ Specific Python/R packages is managed by users via conda
- ✓ Read more about [Python Package Management](#)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



Starting to Compute

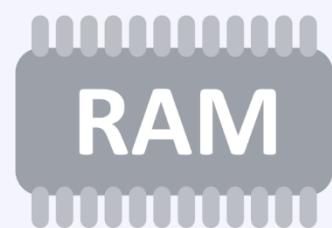
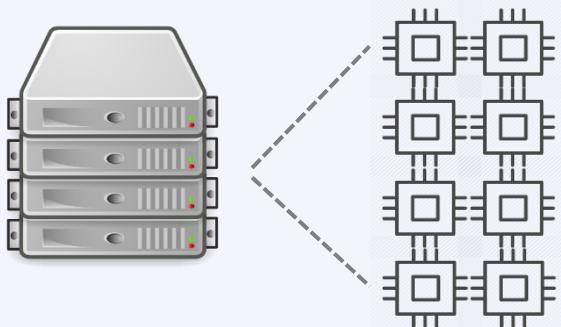
Resource Glossary

- ✓ **Cluster:** Which machine to use? Genius or wlCE?
- ✓ **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

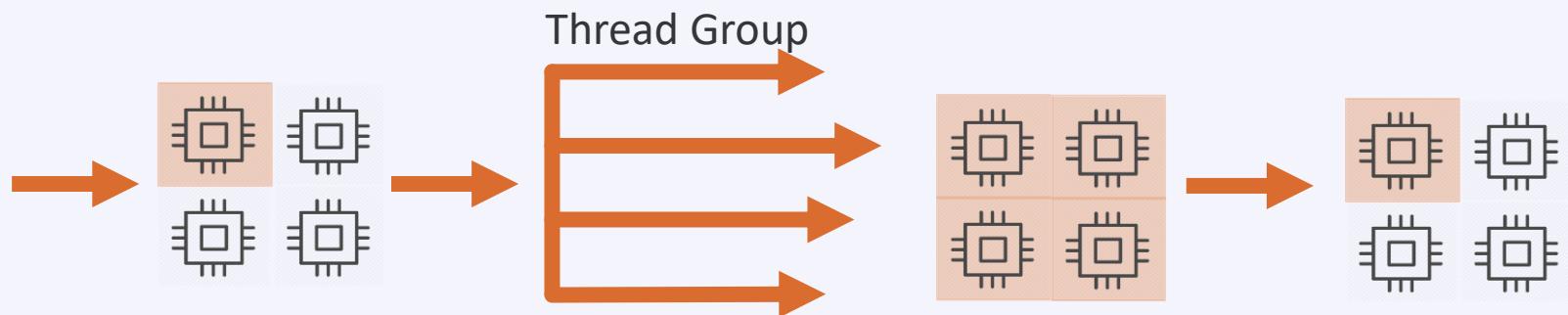
- ✓ **Cluster:** the local machine you are logged into (Genius or wlCE)
- ✓ **Nodes:** 1
- ✓ **Cores:** 1
- ✓ **RAM:** depends on which node(s) you use
- ✓ **Partition:** batch
- ✓ **Walltime:** 1 hour
- ✓ **Storage:** no default
- ✓ **Credits:** no default



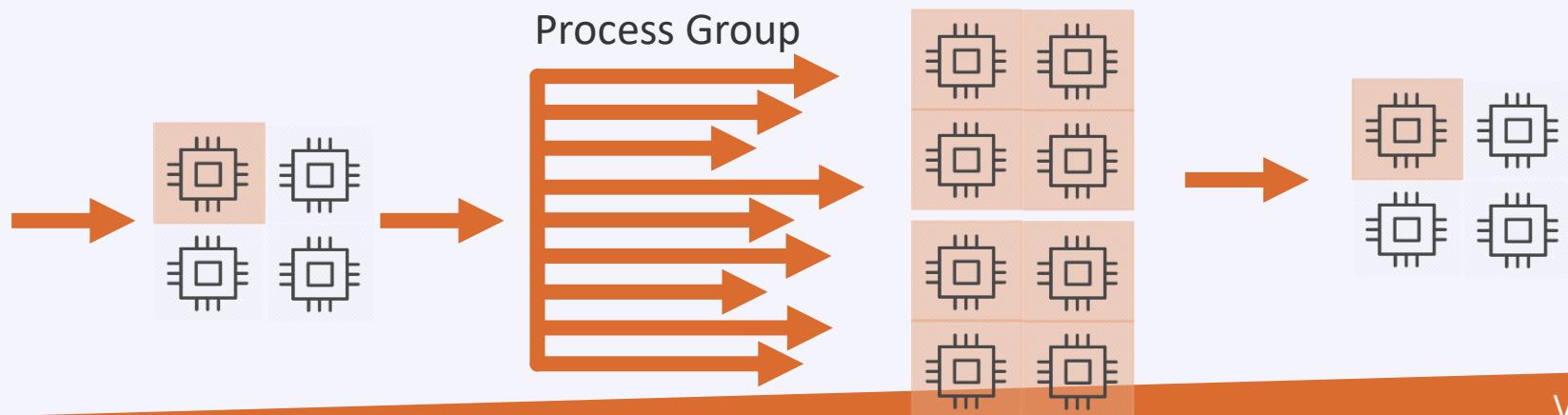
Serial Application (1 process on 1 core)



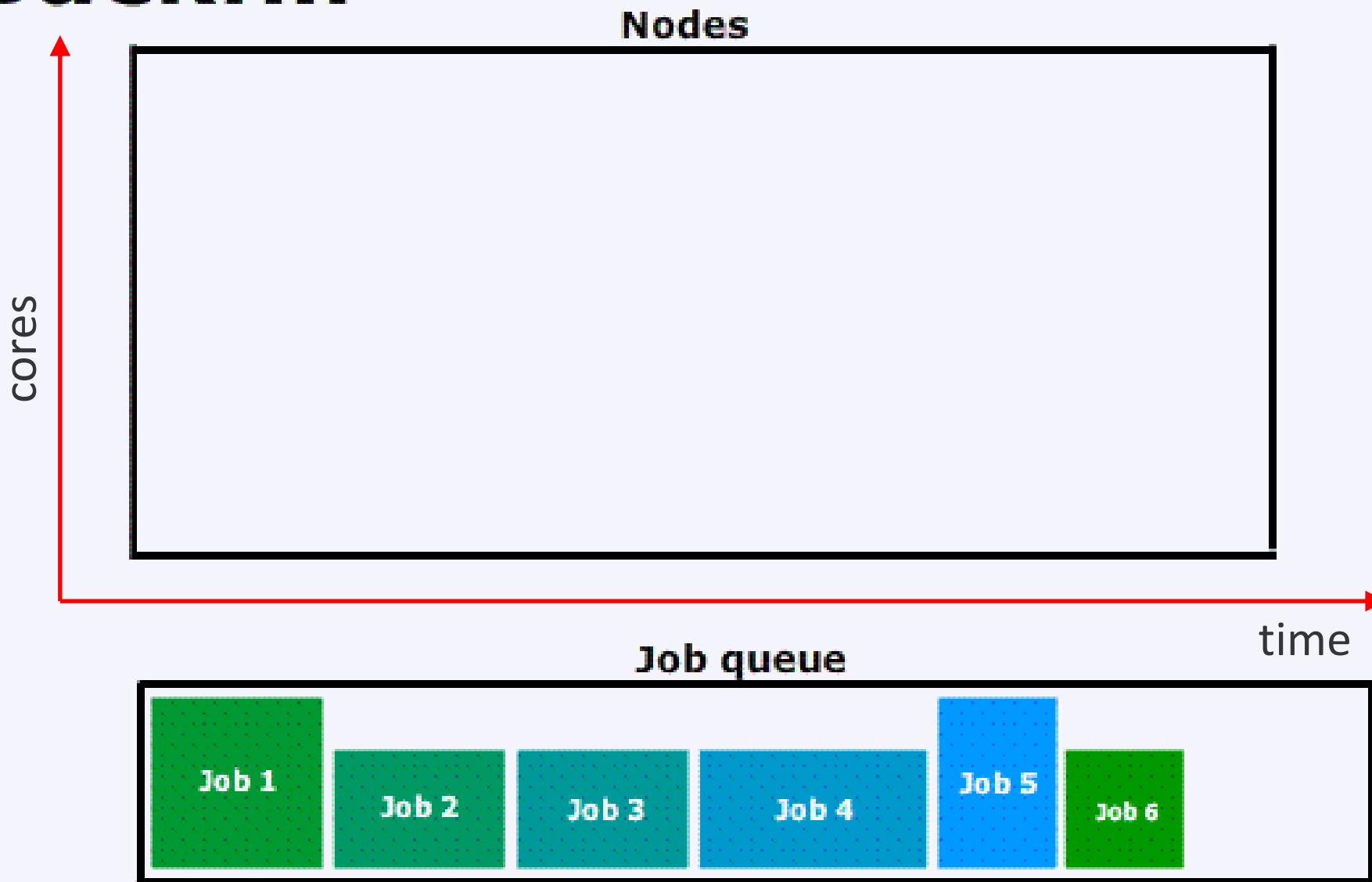
Multi-Core Appl. (N threads on N cores from **1 node**)



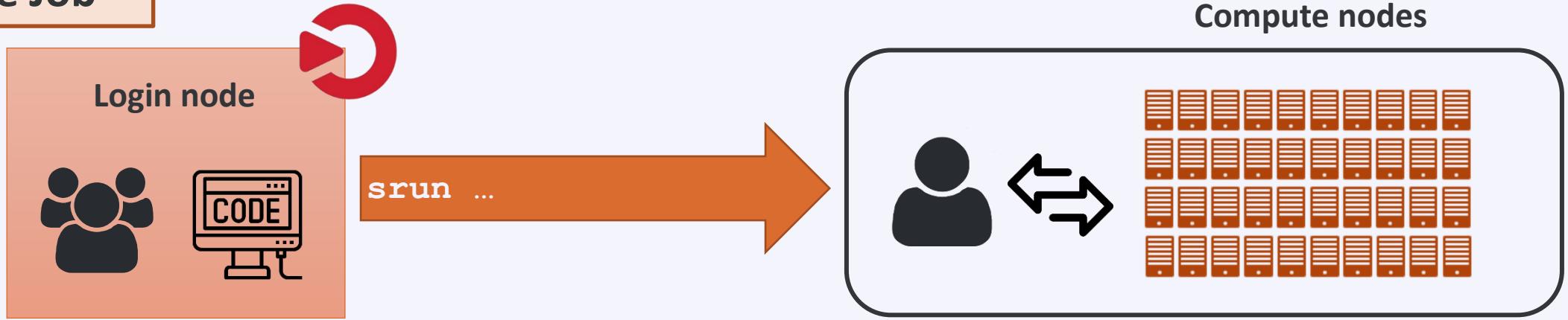
Distributed Appl. (many processes on multiple cores/nodes)



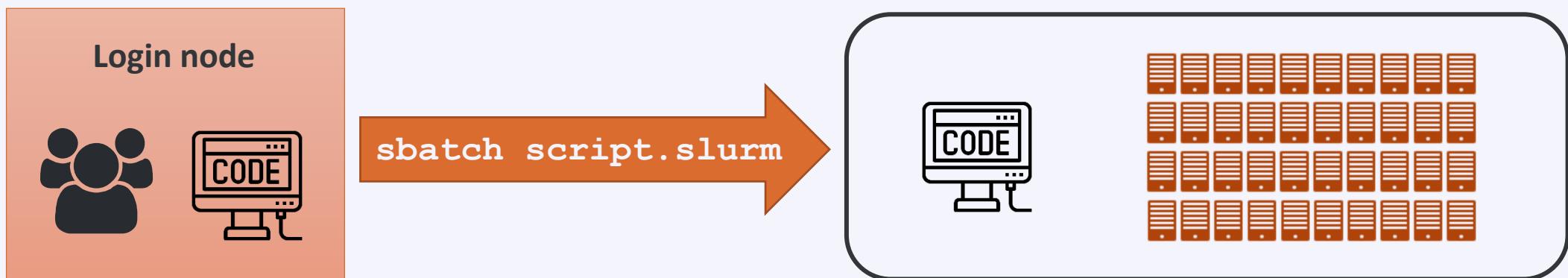
Backfill



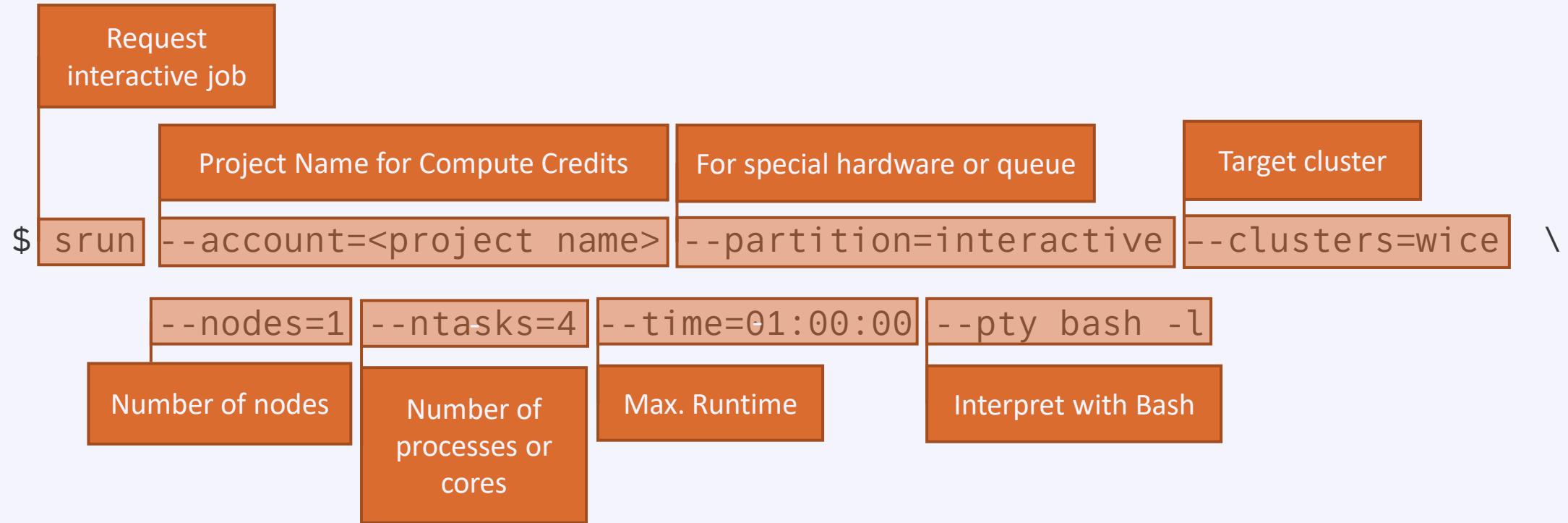
Interactive Job



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=<size><M> | Min memory per core, e.g. 5000M for Genius |
| --time=<dd-hh:mm:ss> | e.g. 1-12:30:00 |
| --job-name=<job_name> | |
| --output=<file_template> | STDOUT; default="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
```

- ✓ To join a running job

```
$ srun --overlap -jobid=<JobID>
```

Example Slurm Job Script

```
#!/bin/bash -l
#SBATCH --clusters=wice
#SBATCH --account=lp_hpcinfo
#SBATCH --nodes=1
#SBATCH --ntasks=4
#SBATCH --mem-per-cpu=3400M
#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
```

Diagram illustrating the components of the Slurm job script:

- Shebang**: `#!/bin/bash -l`
- Resource List**: `#SBATCH` directives
- Module load(s)**: `module load` commands
- Move data**: `cp` command to move input data
- Execute commands**: `python modelling.py`
- Move data**: `cp` and `rm` commands to move output data

Batch Workload

Job Script

```
#!/bin/bash -l  
...  
...
```

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

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: vscXXXXXX
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 or directo
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 wlCE

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

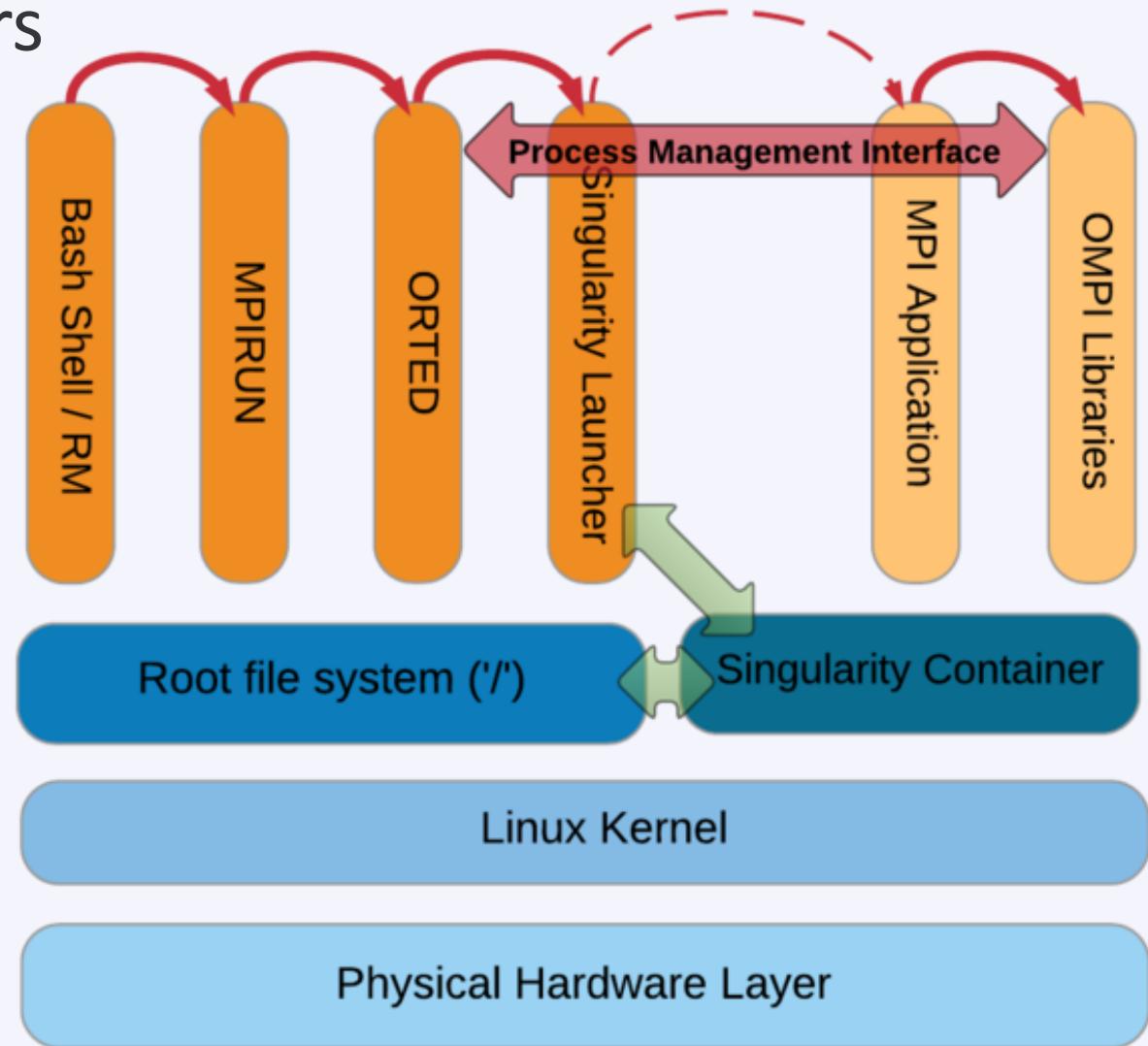
Interactive Job

```
$ 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

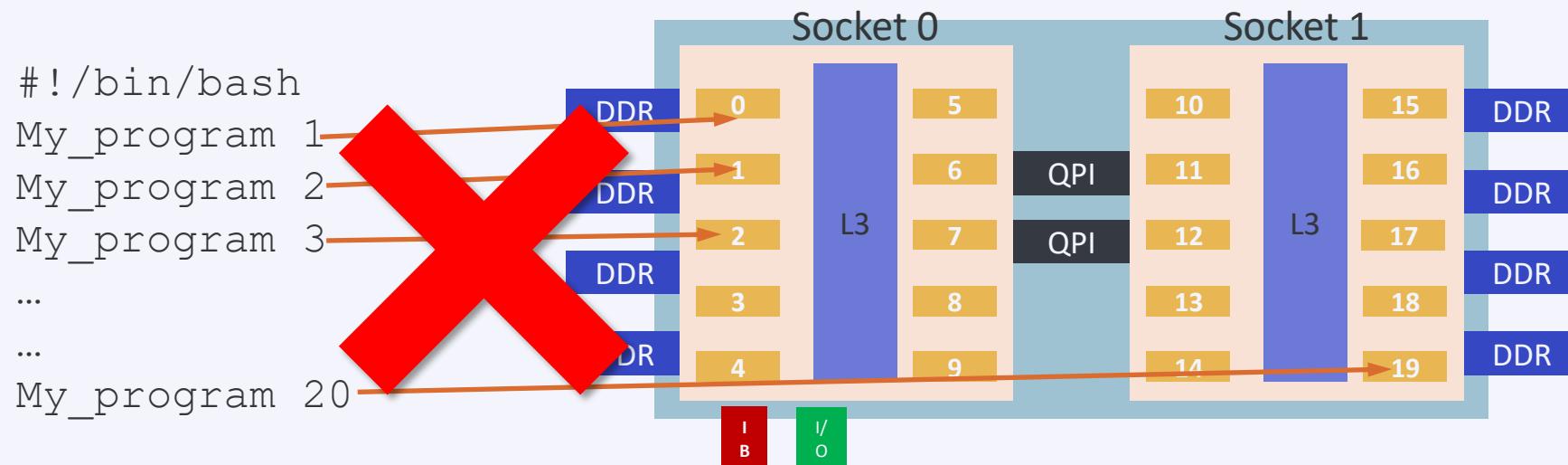
```
#!/bin/bash -l
script.slurm
#SBATCH --time=30:00
#SBATCH --nodes=2
#SBATCH --ntasks=72
#SBATCH --clusters=wice
#SBATCH --account=lp_hpcinfo
apptainer run Project.simg
./model.exe
```



Worker Framework

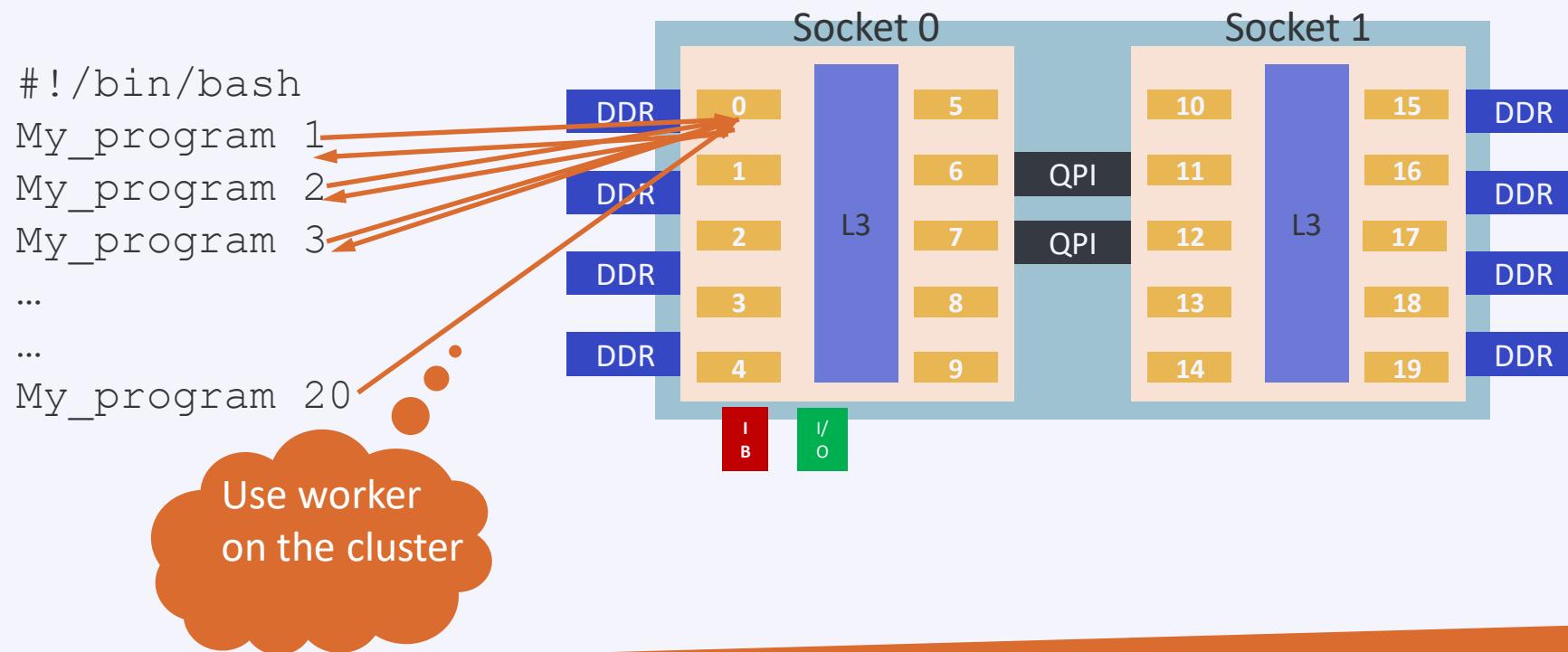
Sequential...

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



Sequential...

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



Parallel Computing

Talk to us about
worker framework

- ✓ 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

Lucky
you!

Use case: parameter exploration

| temperature | pressure | humidity |
|-------------|----------|----------|
| 293.0 | 1.0e05 | 87 |
| ... | ... | ... |
| 313.0 | 1.3e05 | 75 |

```
#!/bin/bash -l
#SBATCH --account=lp_myproject
#SBATCH --cluster=wice
#SBATCH --nodes=1
#SBATCH --ntasks-per-node=1
#SBATCH --time=10:00
cd $SLURM_SUBMIT_DIR
weather -p 1.0e05 -t 293.0 -h 87
```

job_030.slurm

job_600.slurm

Many single core computations

Solution: worker with -data

| temperature | pressure | humidity | data.csv |
|-------------|----------|----------|----------|
| 293.0 | 1.0e05 | 87 | |
| ... | ... | ... | |


```
#!/bin/bash -l
#SBATCH --cluster=genius
#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 Genius login node:
$ module load worker/1.6.12-foss-2021a
$ wsub -data data.csv -batch job.slurm
```

Solution: worker with -data

| temperature | pressure | humidity | data.csv |
|-------------|----------|----------|----------|
| 293.0 | 1.0e05 | 87 | |
| ... | ... | ... | |

```
#!/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
```

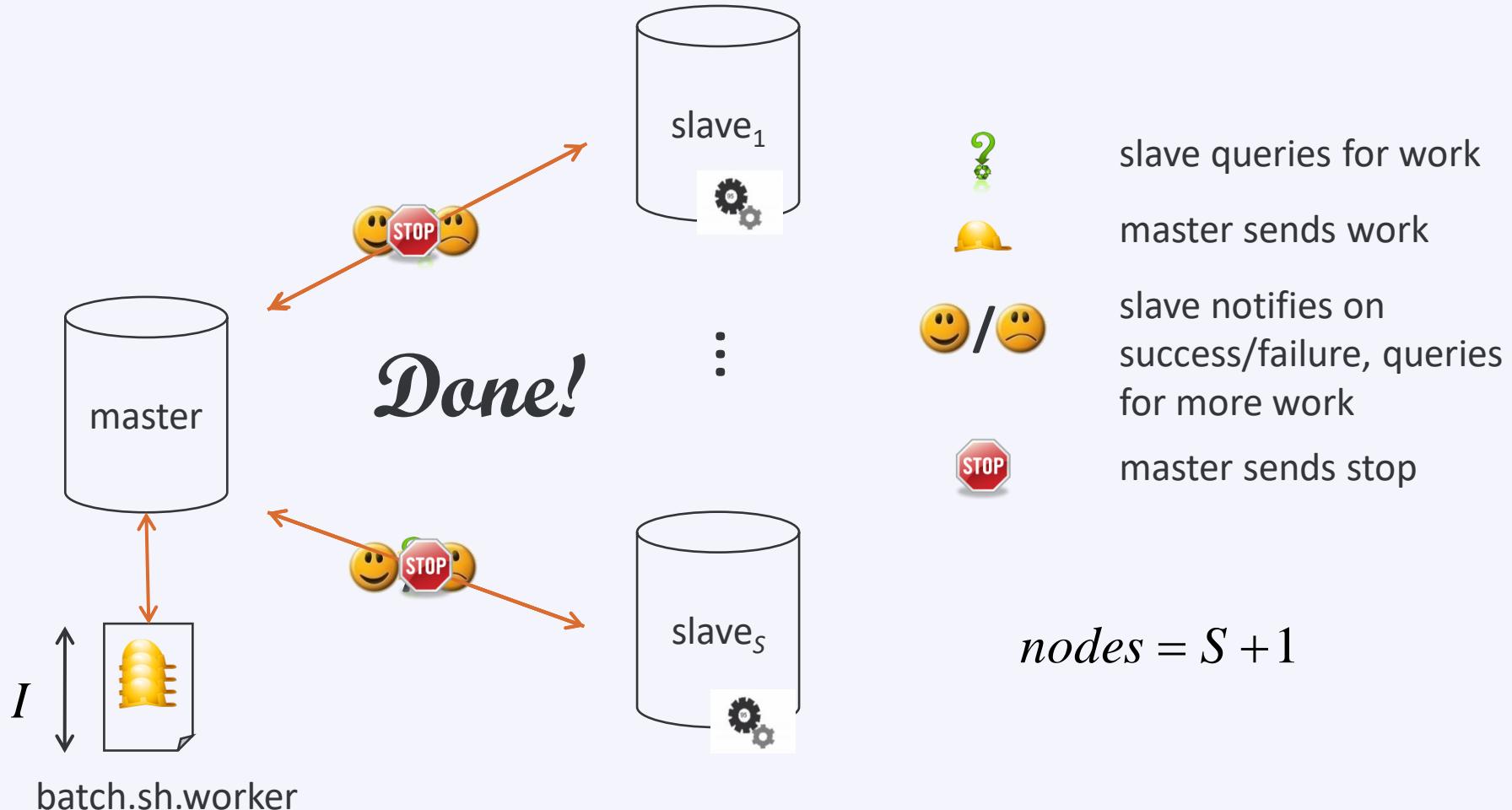
```
# From wICE interactive session:
$ srun --clusters=wice --account=lpt2_sysadmin --
partition=interactive --pty bash -l
$ module load worker/1.6.12-foss-2021a
$ wsub -data data.csv -batch job.slurm
```

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



How to use worker well?

- ✓ Many work items, i.e., $\#work\ items/\#proc \gg 1$
- ✓ 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 ...

```
#!/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

- Let us use one of the worker examples:

https://github.com/gjbex/worker/tree/development_slurm/examples/bash_example

```
#!/bin/bash -l          alp.slurm
#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
```

| letter | counter | alpha.csv |
|--------|---------|-----------|
| 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
[Sat Jul 14 13:07:11 2023] --> [Mon Jul 17 10:44:00 2023]
```

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
```

The background of the slide features a vibrant, colorful illustration of a train station. On the left, a train with several green and brown cylindrical cargo containers is visible. In the center, a large digital screen displays a grid of numbers, possibly a train schedule or a financial chart. To the right, a train with blue and red cylindrical cargo containers is shown. The overall scene is dynamic and modern.

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*

