ETH zürich

Scientific IT Services
High Performance Computing

Getting started with the scientific cluster

Samuel Fux, Nadia Marounina
High Performance Computing Group
Scientific IT Services, ETH Zurich

Outlook

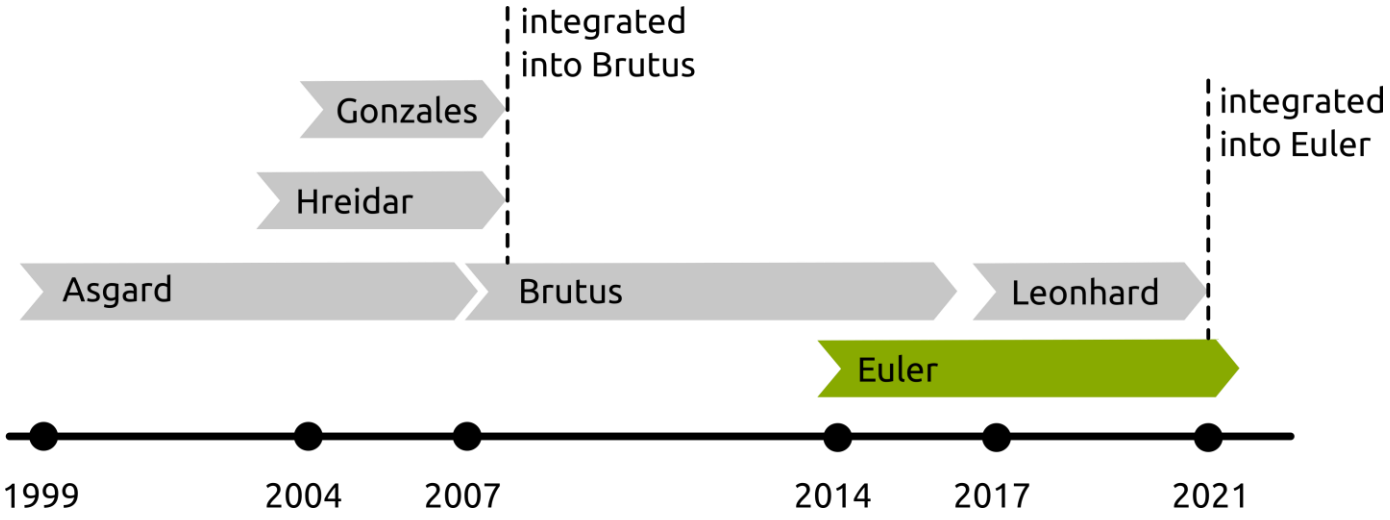
- Introduction
- Accessing the cluster
- Storage and data transfer
- Modules and applications

Outlook

- Introduction
- Accessing the cluster
- Storage and data transfer
- Modules and applications

Intro > What is EULER ?

- EULER stands for **E**rweiterbarer, **U**mweltfreundlicher, **L**eistungsfähiger **E**TH Rechner (Scalable, environment-friendly, powerful ETH Computer)
- Benefits from 20 years of experience with HPC clusters at ETH



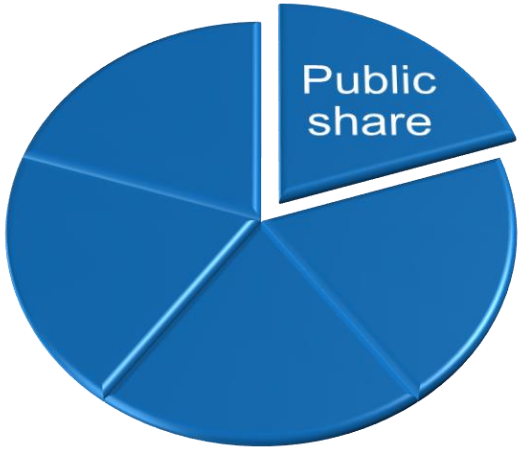
Intro > Shareholder model

Shareholders

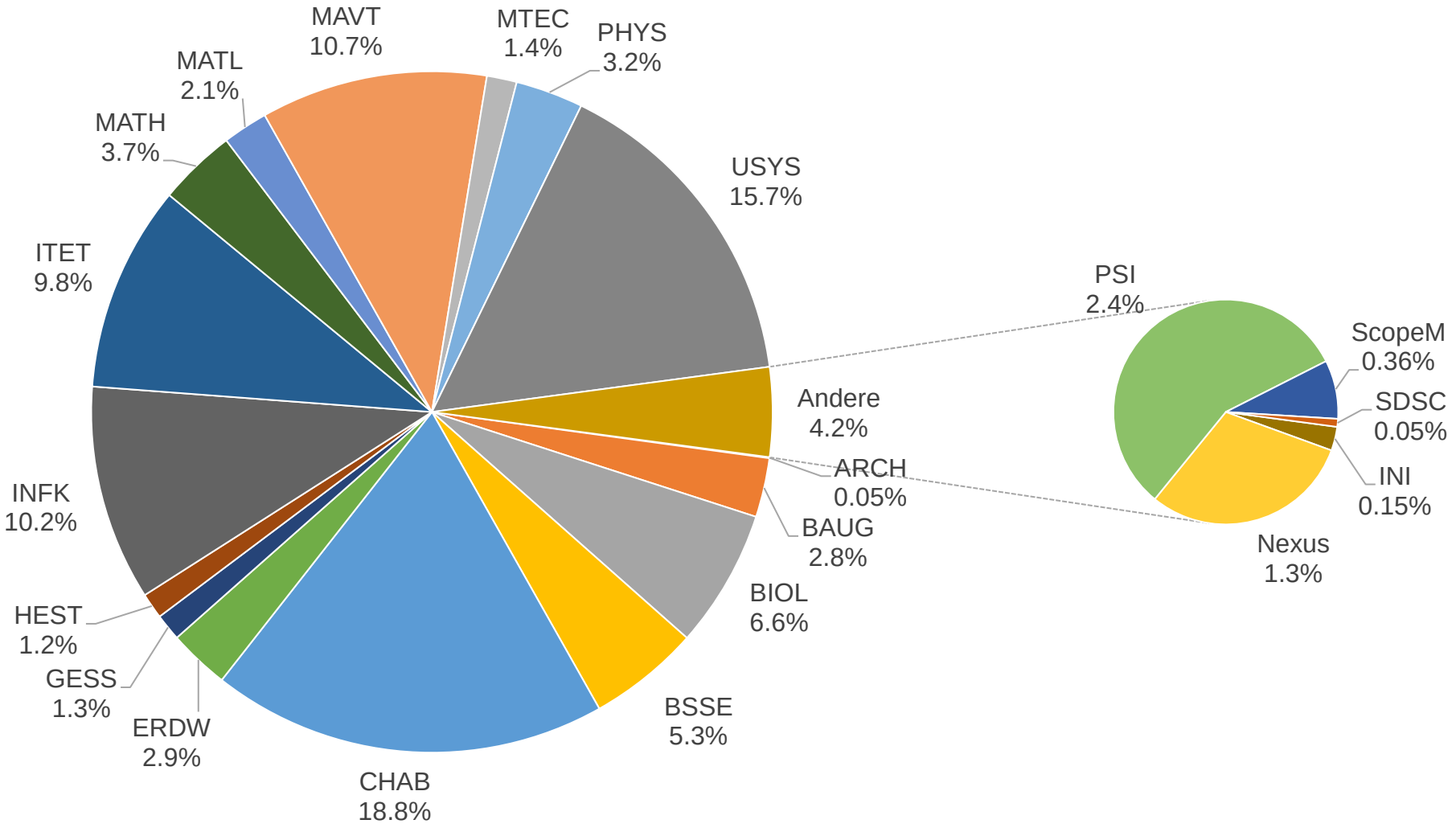
- Like its predecessors, Euler has been financed (for the most part) by its users
- So far, over 180 (!) research groups from almost all departments of ETH have invested in Euler
- These so-called **shareholders** receive a share of the cluster's resources (processors, memory, storage) proportional to their investment

Public share

- A small share of Euler financed by IT Services is open to all members of ETH
- The only requirement is a valid ETH account
- These **guest users** can use limited resources
- If someone needs more computing power, they can invest in the cluster and become a shareholder at any time

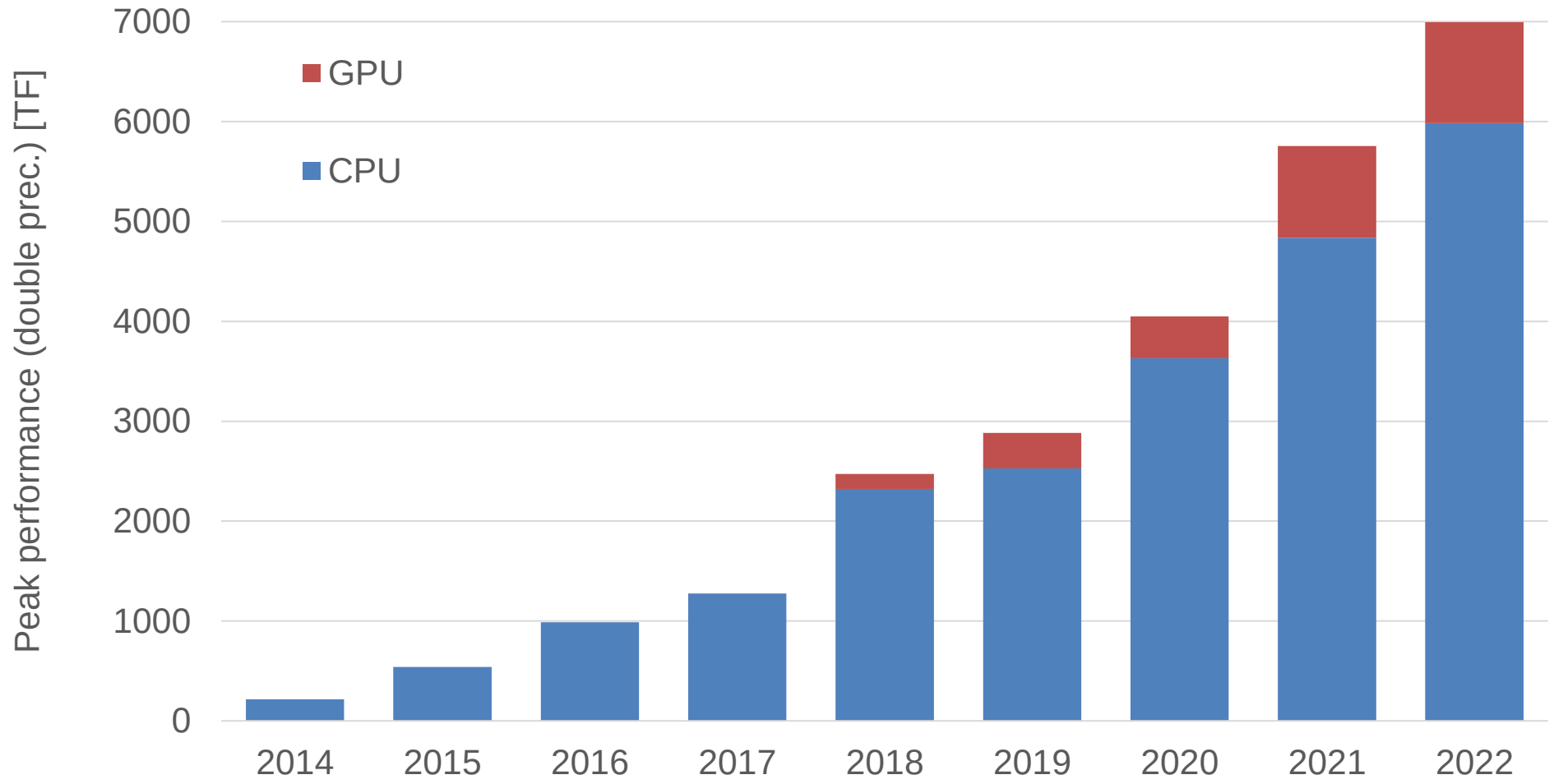


Intro > Euler shares by departments (February 2022)



Intro > Performance of the Euler cluster

All values correspond to the state in March of each year
Peak performance in double precision based on CPU/GPU nominal frequency; effective performance is lower



Intro > Euler II (left) & Euler V (right)

Preparation > Linux terminal > Bash

- Bash is a powerful scripting language, which can directly be used in a Linux terminal and is often combined with Linux commands

```
[sfux@eu-login-01 ~]$ for i in *; do if [ -d $i ]; then echo $i; cd $i; du -hs; cd ..; fi; done
bin
52K     .
lib64
36K     .
scripts
192K    .
test
324M    .
testrun
693M    .
[sfux@eu-login-01 ~]$
```

https://tldp.org/LDP/Bash-Beginners-Guide/html/

Preparation > Linux terminal > Basic commands

Command	Explanation
`ls`	Directory listing
`cd`	Change directory
`pwd`	Print working directory
`echo`	Print to standard output
`less, cat, more, head, tail`	Display content of a file to standard output
`cp, mv`	Copy/move a file or directory
`rm`	Remove a file or directory
`mkdir`	Create a directory
`vi, nano, emacs`	Command line text editor
`man`	Show manual for a command in terminal
`grep`	Search for a pattern in a string or file

Preparation > Linux terminal > Basic commands

Command	Explanation
`exit`	Terminate and exit the current terminal
`sort`	Sort a file line by line
`sed`	String manipulation
`awk`	Programming language for text processing
`find`	Search for files
`du`	Show disk space used in a directory
`tar`	Create a tar archive with files/directories
`gzip`	Compress files/directories
`top`	Real time view of processes in a computer
`chmod`	Change permissions of file/directory

https://scicomp.ethz.ch/wiki/Linux_command_line

Preparation > Linux permissions

- In Linux, access to files and directories is handled via permissions
  - Read permission (r) - grants permission to read a file or directory
  - Write permission (w) - grants permission to write a file or directory
  - Execute permission (x) - grants permission to execute a file or directory
- There are 3 permission groups
  - User (u) - permissions for the user account that owns the file
  - Group (g) - permissions for the user group that owns the file
  - Other (o) - permissions for all other users except the user account and the user group

```
[sfux@eu-login-29 ~]$ ls -l gurobi.log
-rw-r--r-- 1 sfux sfux-group 800 Sep 17 10:29 gurobi.log
[sfux@eu-login-29 ~]$ ls -ld data
drwxr-xr-x 2 sfux sfux-group 4096 Jan  9  2017 data
```

https://scicomp.ethz.ch/wiki/Linux_permissions

Preparation > Linux permissions

- Another method for representing Unix permissions is an octal (base 8) notation
  - The read bit adds 4 to its total (in binary 100),
  - The write bit adds 2 to its total (in binary 010), and
  - The execute bit adds 1 to its total (in binary 001).
  - Example: `r-x` → 101 → 4+0+1=5
  - (u), (g) and (o) are then combined (755 represents `rwxr-xr-x`)
- Permissions can be changed with the `chmod` command
  - String representation:

    ```
    $ chmod ugo+rx filename
    ```

  - Number representation:

    ```
    $ chmod 750 filename
    ```

https://en.wikipedia.org/wiki/File-system_permissions#Numeric_notation

Preparation > Paths > Linux/Mac OS X vs. Windows

- There are differences how operating systems are representing file paths
- Paths on Windows: `C:\Users\Samfux\test`
  - Use backslashes
  - File and directory names are not case sensitive
  - Different drives (`C:`, `D:`, etc.)
- Paths on Linux/Mac OS X: `/cluster/home/sfux/test`
  - Use forward slashes
  - Everything is case sensitive
  - Everything is under the root file system (`/`), no drives

Preparation > Edit a text file with vim

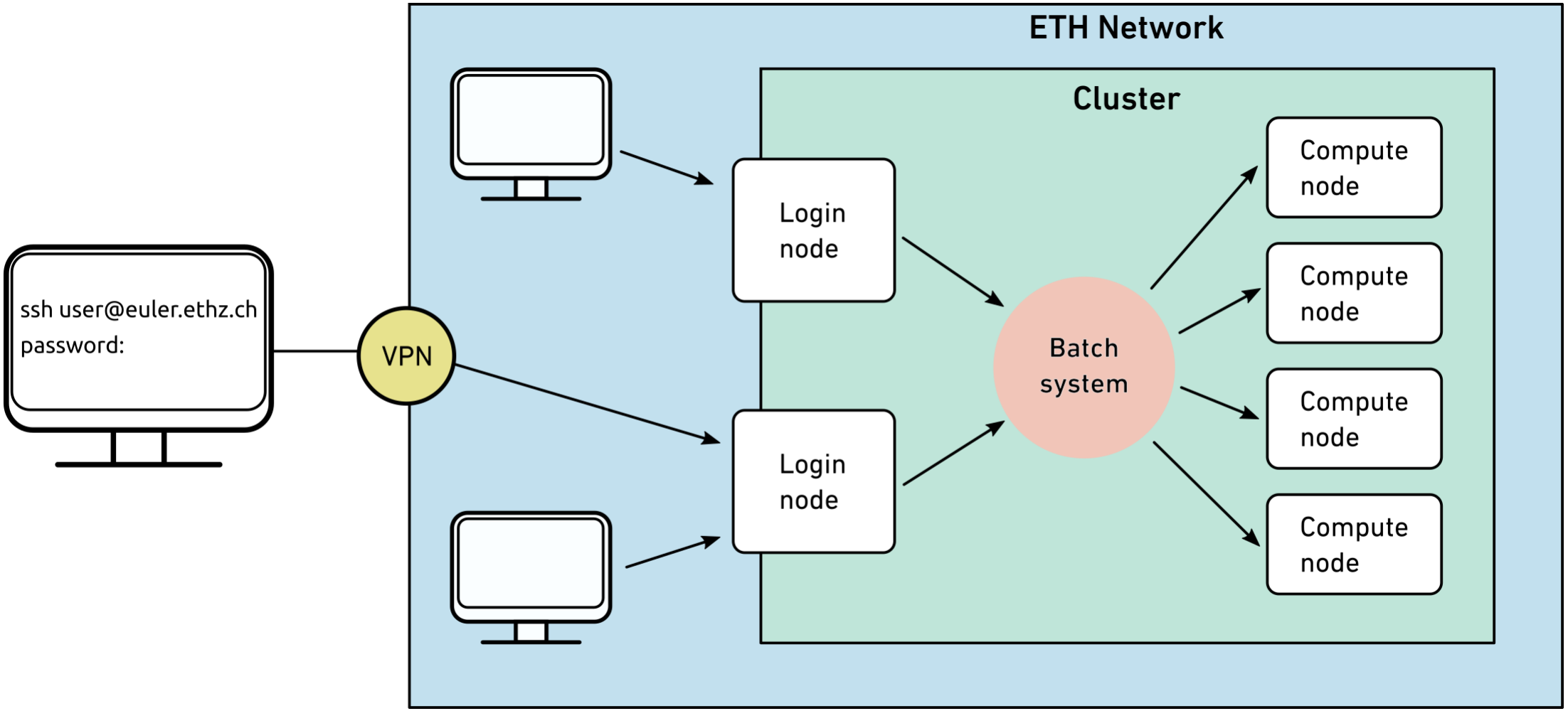
- Vim is a command line text editor that can be started with the command `vi`. This text editor is useful to edit input files and write scripts directly on the cluster without copying forth and back files
- 2 modes (insert mode, command mode)
  - Type `i` to switch from command mode to insert mode
  - Type `esc` to switch from insert mode to command mode
- Insert mode: You can insert text into a text document
- Command mode: You can type commands after typing `:` (colon) and execute it with the enter key
  - To save a file, type `:w`
  - To close a file, type `:q`
  - To save and close a file, type `:wq`

https://www.tutorialspoint.com/unix/unix-vi-editor.htm

Outlook

- Introduction
- Accessing the cluster
- Storage and data transfer
- Modules and applications

Accessing the cluster

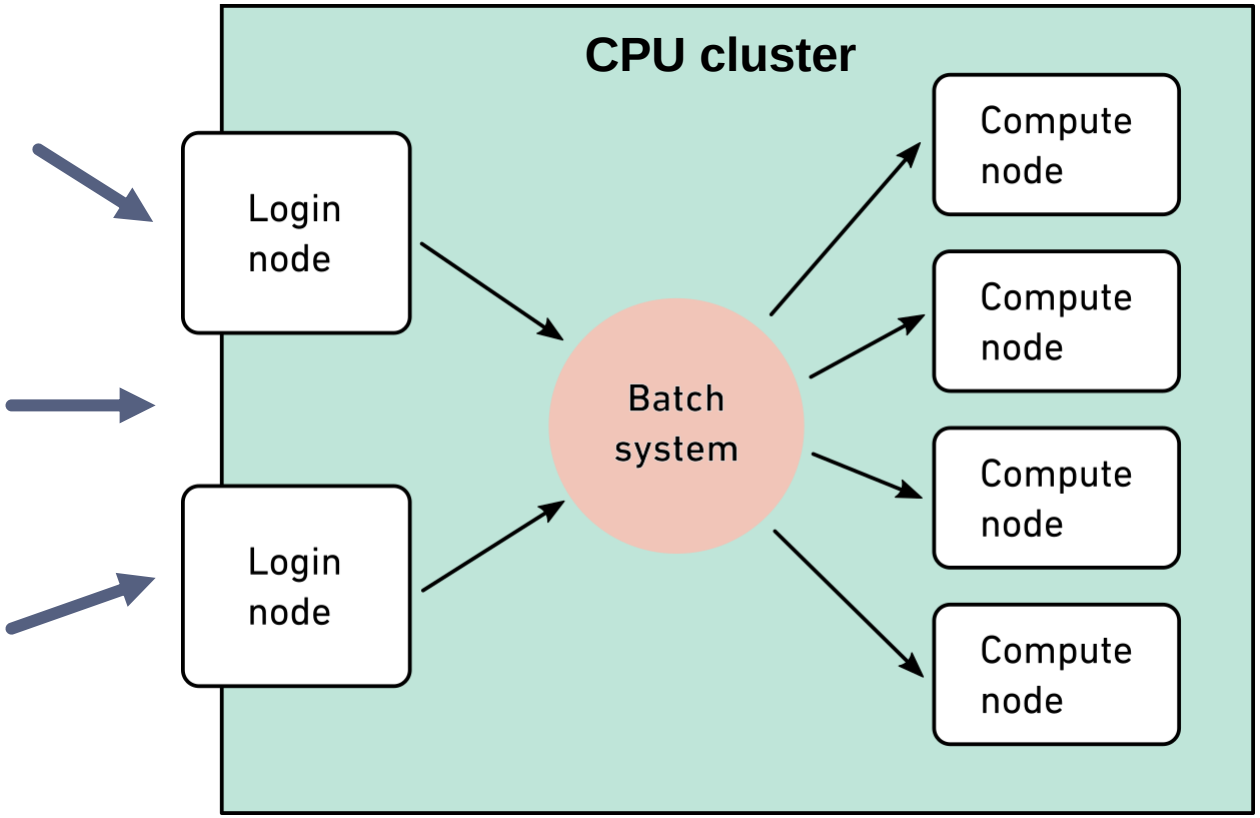


Access > Who can use the cluster > CPU cluster

Shareholders that invested into a share of the cluster resources

External collaboration partners of shareholders

All ETH members (they can use the cluster as guest users with limited resources)

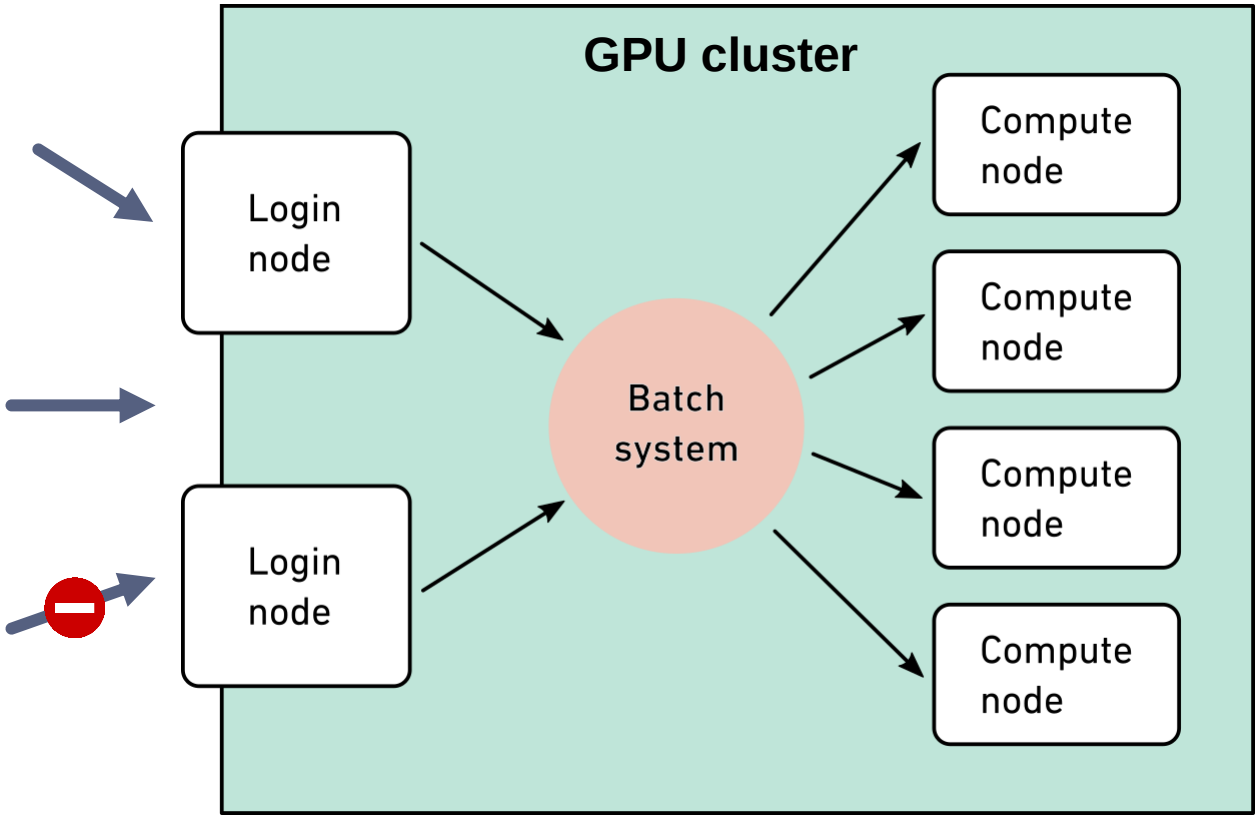


Access > Who can use the cluster > GPU cluster

Shareholders that invested into a share of the cluster resources

External collaboration partners of shareholders

Guest users cannot access the GPU cluster



Access > Prerequisites

- A valid ETH account
- Local computer with an SSH client
  - Linux and macOS contain SSH client as part of the operating system
  - Windows users need to install a third party SSH client
    - MobaXterm (https://mobaxterm.mobatek.net/) is a free open source SSH client that we recommend

- An X11 server for graphical user interface (optional)
  - Linux (https://www.xorg.com)
  - macOS (https://xquartz.org)
  - Windows (included in MobaXterm)

Access > How to access the clusters > ETH members

1. Start your SSH client
2. Use `ssh` command to connect to the login node of Euler

```
ssh username@euler.ethz.ch
```

3. Use your ETH credentials to login
4. First login
   - On first login a verification code is sent to your email address (*username*@ethz.ch)
   - By entering the verification code, your account is created automatically
   - New users must accept the cluster's usage rules

https://scicomp.ethz.ch/wiki/New_account_request_process_for_HPC_clusters

Access > How to access the clusters > External collaborators

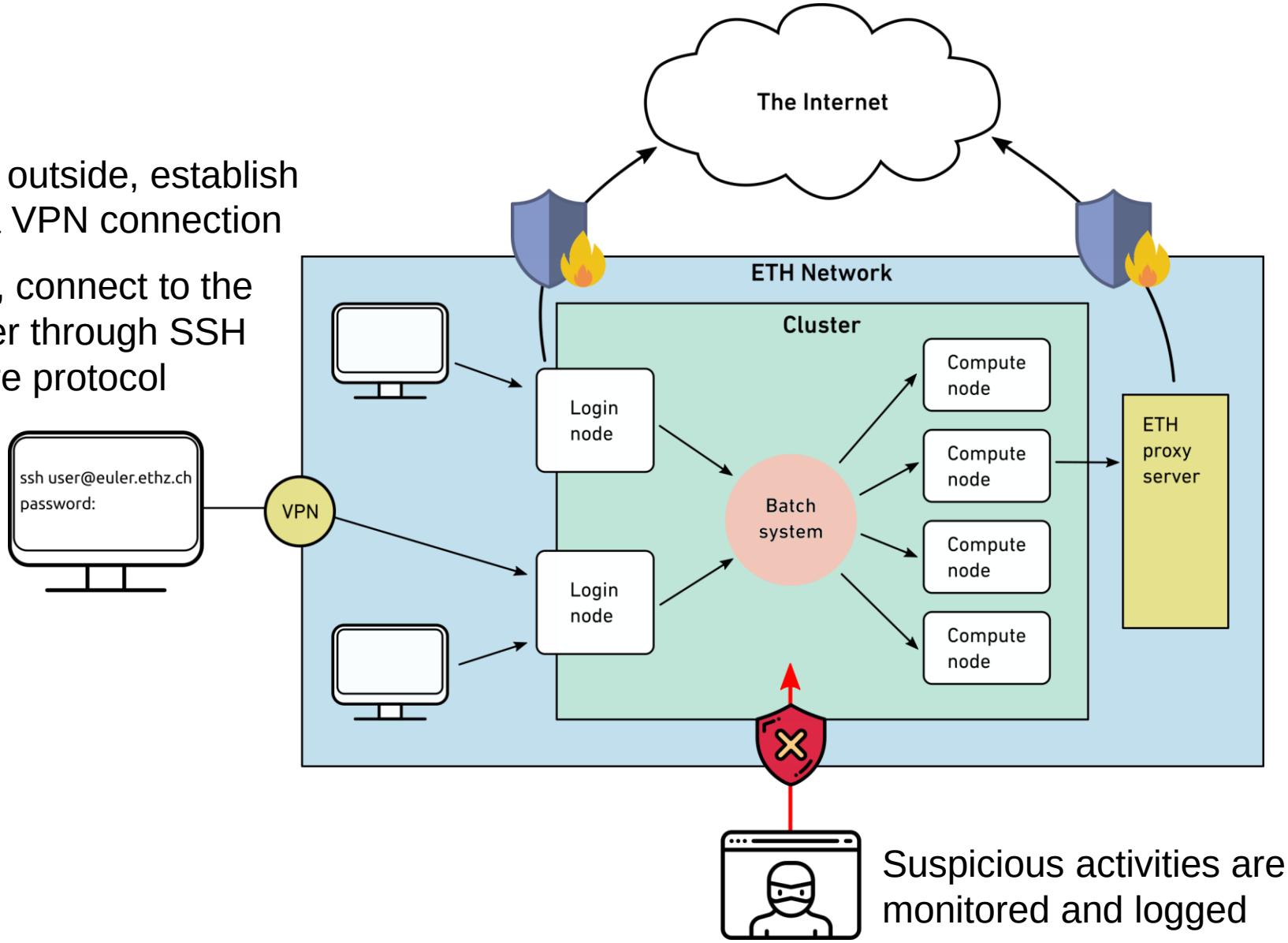
Members of other institutions who collaborate with a research group at ETH may use the clusters for the purpose of said collaboration. To get access:

1. Their ETH counterpart (“sponsor”) create an ETH guest account, including e-mail address and VPN service, for them
2. Access the cluster like ETH members

Access > Security > Firewall

From outside, establish first a VPN connection

Then, connect to the cluster through SSH secure protocol



From a compute node to an external service, use the ETH proxy

```
$ module load eth_proxy
```

Suspicious activities are monitored and logged

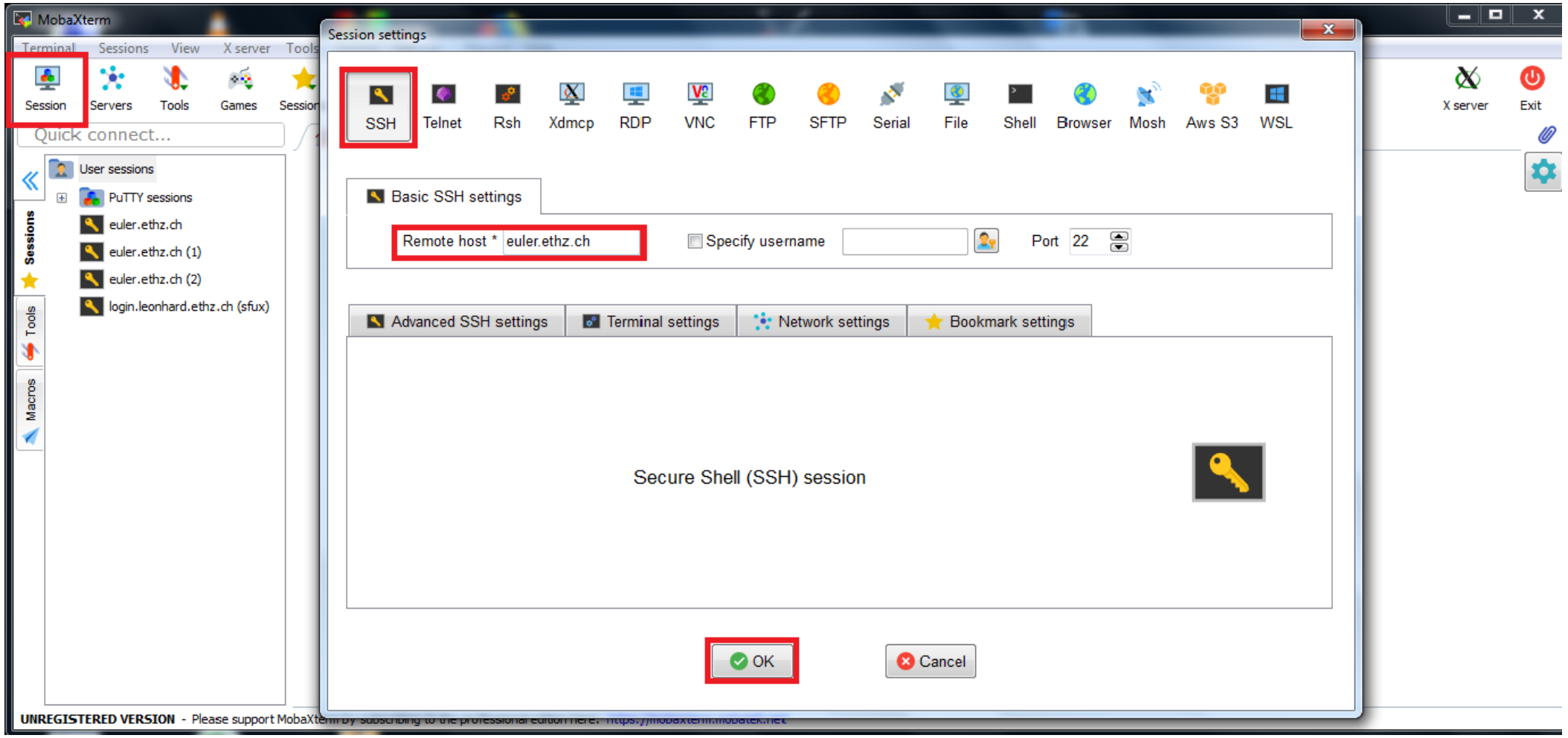
Access > Legal compliance

- The HPC clusters are subject to ETH's acceptable use policy for IT resources (Benutzungsordnung für Telematik, BOT, https://rechtssammlung.sp.ethz.ch/Dokumente/203.21en.pdf), in particular:
  - Cluster accounts are **strictly personal**
  - DO NOT share your account (password, ssh keys) with anyone
  - DO NOT use someone else's account, even if they say it's OK
  - If you suspect that someone used your account:
    - change your password at https://password.ethz.ch
    - contact cluster-support@id.ethz.ch
- Consequences
  - In case of abuse, the offender's account may be blocked temporarily or closed
  - System administrators are obliged by law to investigate abusive or illegal activities and report them to the relevant authorities

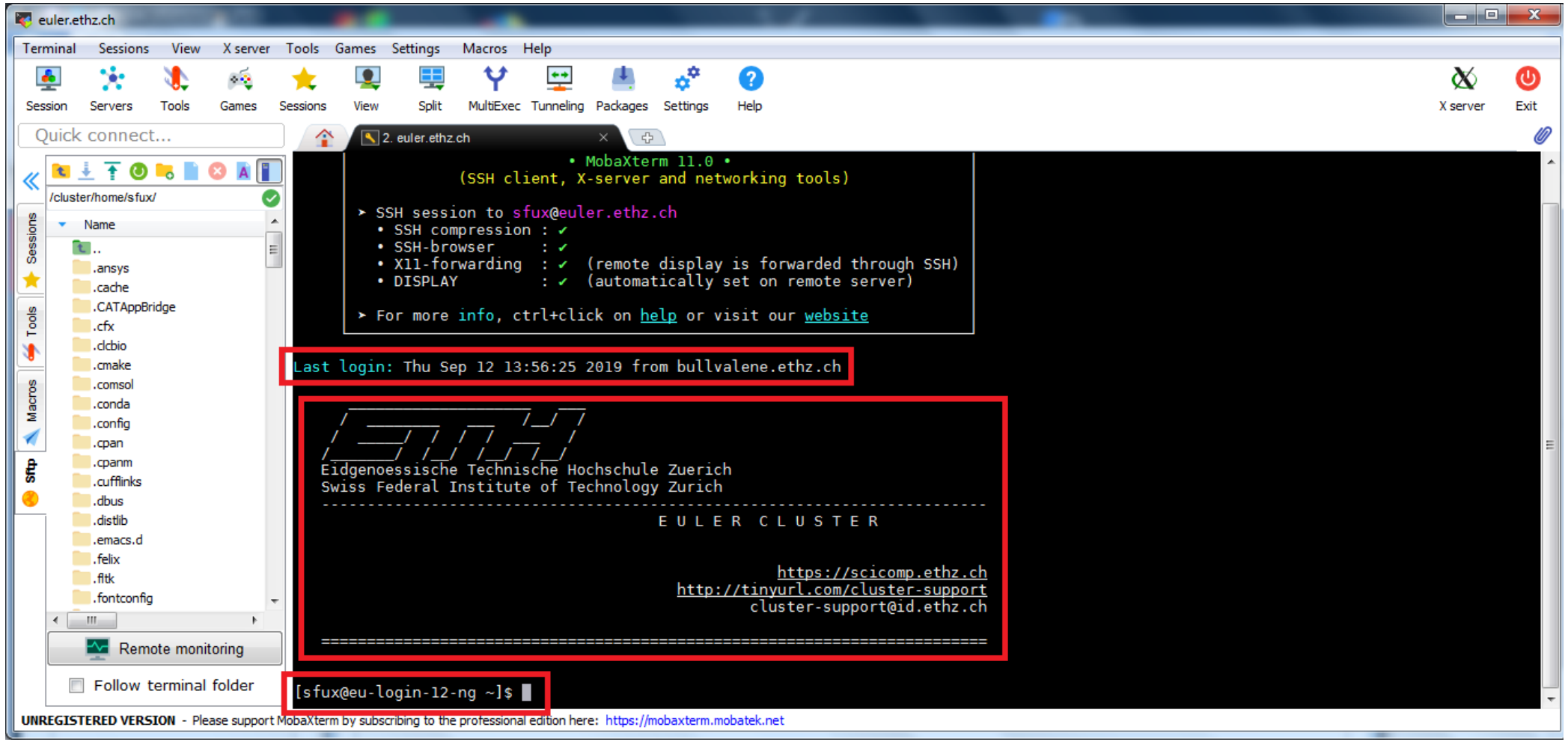
Access > SSH connection > Linux, macOS

```
samfux@bullvalene:~$ ssh sfux@euler.ethz.ch
sfux@euler.ethz.ch's password:
Last login: Fri Sep 13 07:33:57 2019 from bullvalene.ethz.ch

      ____________________   ___
     /  _________   ___   /__/  /
    /  /____/  /  /  /  /___/  /
   /_______/  /__/  /__/  /__/
   Eidgenoessische Technische Hochschule Zuerich
   Swiss Federal Institute of Technology Zurich
   ------------------------------------------------------------------------
                                    E U L E R  C L U S T E R


                                                    https://scicomp.ethz.ch
                                                http://www.smartdesk.ethz.ch
                                                  cluster-support@id.ethz.ch

   ========================================================================

[sfux@eu-login-19 ~]$
```

Access > SSH connection > Windows



Access > SSH connection > Windows



Access > Graphical user interface

Our clusters use the X Window System (X11) to display a program's graphical user interface (GUI) on your workstation

X11 server	Linux	macOS	Windows
Installation	Xorg Usually installed	XQuartz Installation needed	Included in MobaXterm
Usage	Use the log-in command `ssh -Y username@euler.ethz.ch`		Automatically enabled in MobaXterm

Access > SSH keys (Optional)

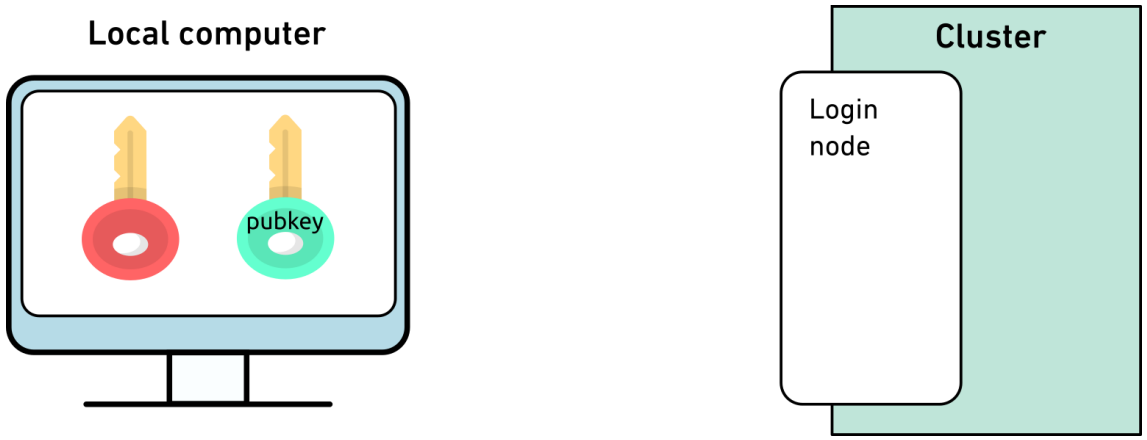
It is not compulsory to create and use SSH keys to participate the cluster workshop.

- SSH keys allows passwordless login
  - Useful for file transfers and automated tasks
  - When used properly, SSH keys are much safer than passwords

- SSH keys always come in pairs
  - A **private** key, stored on your local workstation (and nowhere else!)
  - A **public** key, stored on the computer(s) you want to connect to

- You can generate as many pairs as you like, e.g., one for each computer you intend to connect to
- Keys should be protected with a passphrase
- SSH key management tools such as `ssh-agent` and `keychain` help unlock SSH keys

Access > SSH keys > Step 1: Create your keys

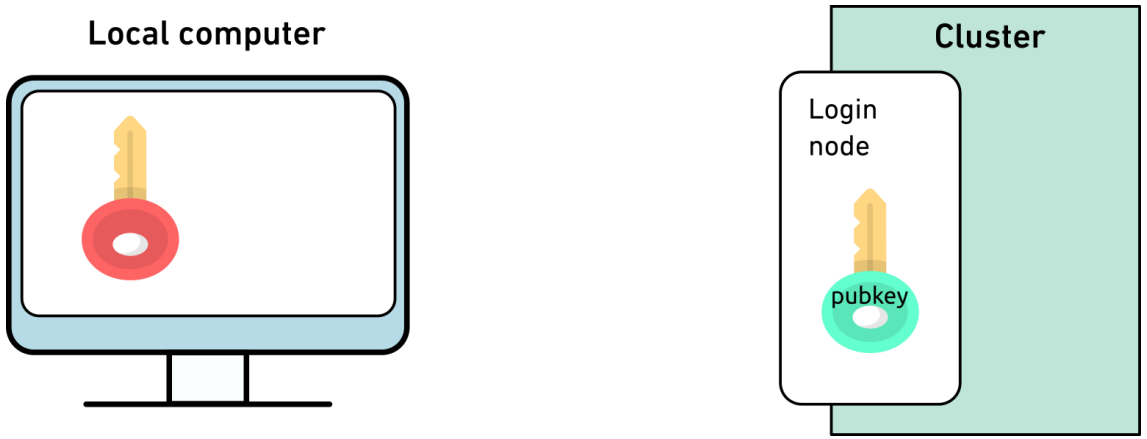


- Verify whether logging in with password works
- Generate a key pair with the ed25519 algorithm for each computer you want to connect to

```
ssh-keygen -t ed25519 -f $HOME/.ssh/id_ed25519_euler
```

- Enter a passphrase to protect your SSH keys

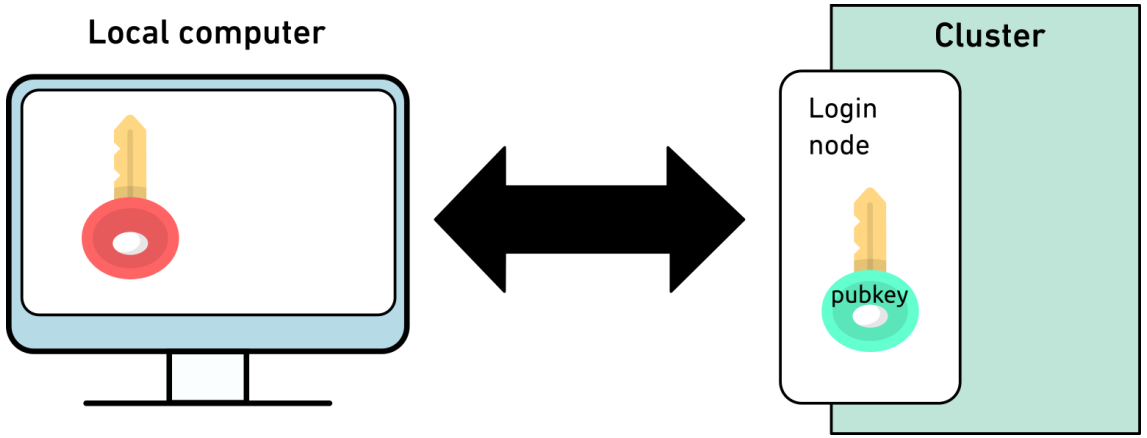
Access > SSH keys > Step 2: Copy the public key to the cluster




```
ssh-copy-id -i $HOME/.ssh/id_ed25519_euler.pub    username@euler.ethz.ch
```

https://scicomp.ethz.ch/wiki/Accessing_the_clusters#SSH_keys

Access > SSH keys > Step 3: Use keys with non-default names




```
ssh -i $HOME/.ssh/id_ed25519_euler    username@euler.ethz.ch
```

https://scicomp.ethz.ch/wiki/Accessing_the_clusters#How_to_use_keys_with_non-default_names

Access > SSH keys > Use keys with non-default names

- SSH clients can use this option automatically by adding the option `IdentityFile` in your `$HOME/.ssh/config` file, e.g.:

```
Host euler
HostName euler.ethz.ch
User username
IdentityFile ~/.ssh/id_ed25519_euler
IdentitiesOnly yes
```

- Next time you login, you can type

```
$ ssh euler
```

https://scicomp.ethz.ch/wiki/Accessing_the_clusters#How_to_use_keys_with_non-default_names

Access > SSH Key Management > SSH Agent

As we have to enter the passphrase to unlock the keys, it takes away the convenience of passwordless login. We can use an SSH agent (`ssh-agent`) to unlock the SSH keys per terminal on some distributions.

```
$ eval `ssh-agent`
Agent pid 17906

$ ssh-add -l
The agent has no identities.

$ ssh-add $HOME/.ssh/id_ed25519_euler
Enter passphrase for id_ed25519_euler:
Identity added: id_ed15519_euler (username@localcomputer)
```

Access > Exercise 1: Create SSH keys

Tasks	Commands
On your local computer, create SSH keys	`ssh-keygen -t ed25519 -f $HOME/.ssh/id_ed25519_euler`
Check the keys	`ls -l $HOME/.ssh`
Copy the key to the cluster	`ssh-copy-id -i $HOME/.ssh/id_ed25519_euler.pub` *`username`*`@euler.ethz.ch`
Log in with the non-default name key	`ssh –i $HOME/.ssh/id_ed25519_euler` *`username`*`@euler.ethz.ch`
Exit Euler	`exit`

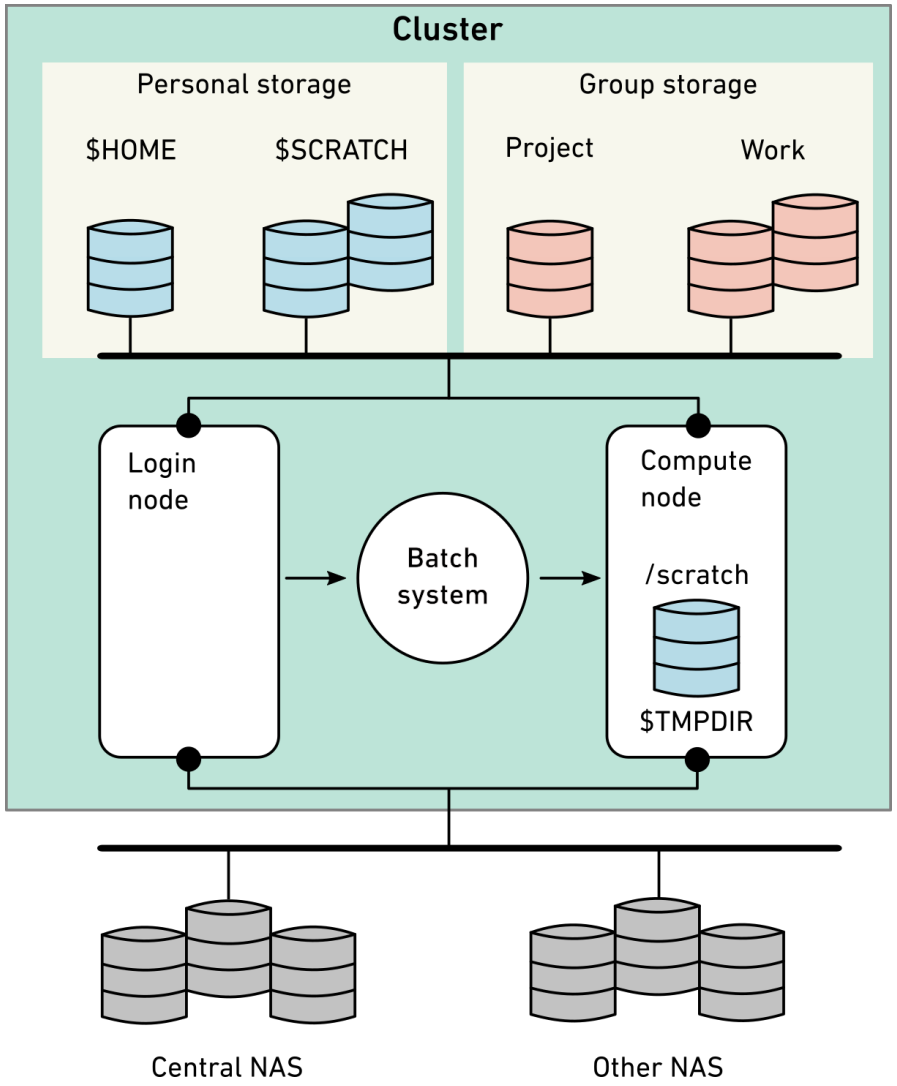
Access > Exercise 2

Tasks	Commands
Log in to Euler	`ssh username@euler.ethz.ch`
Check your current directory	`pwd`
Go to your scratch folder	`cd $SCRATCH`
Check current loaded modules	`module list`
Submit a job	`sbatch --wrap="sleep 120"`
Check job status	`squeue`
After the job has finished, check the output file	`cat slurm-*`

Outlook

- Introduction
- Accessing the cluster
- Storage and data transfer
- Modules and applications

Data > Available storage systems



- Cluster wide storage systems
  - Home (personal)
  - Global scratch (personal)
  - Work (group)
  - Project (group)
- Local storage inside the compute node
  - Local scratch
- External storage
  - NAS
  - CDS
  - LTS

Data > Personal storage (every user) > Home

```
$ cd $HOME
$ pwd
/cluster/home/username
```

- Safe, long-term storage for critical data (program source, scripts, etc.)
- Accessible only by the user (owner); other people cannot read its contents
- Disk quota of 16/20 GB and a maximum of 80’000/100’000 files (soft/hard quota). Quota can be checked with the command `lquota`
- Contents saved every hour/day/week using snapshot. Users can access these snapshots in the hidden `.snapshot` directory

https://scicomp.ethz.ch/wiki/Storage_systems#Home

Data > `lquota`

```
[sfux@eu-login-02 ~]$ lquota
+----------------------+------------+-----------+---------------+---------------+
| Storage location:    | Quota type: | Used:     | Soft quota:   | Hard quota:   |
+----------------------+------------+-----------+---------------+---------------+
| /cluster/home/sfux   | space      |    8.85 GB |      17.18 GB |      21.47 GB |
| /cluster/home/sfux   | files      |      25610 |        160000 |        200000 |
+----------------------+------------+-----------+---------------+---------------+
| /cluster/shadow      | space      |    4.10 kB |       2.15 GB |       2.15 GB |
| /cluster/shadow      | files      |          2 |         50000 |         50000 |
+----------------------+------------+-----------+---------------+---------------+
| /cluster/scratch/sfux | space      |  237.57 kB |       2.50 TB |       2.70 TB |
| /cluster/scratch/sfux | files      |         29 |       1000000 |       1500000 |
+----------------------+------------+-----------+---------------+---------------+
```

Data > Personal storage (every user) > Global scratch vs. local scratch

Global scratch

```
$ cd $SCRATCH
$ pwd
/cluster/scratch/username
```

- Fast, short-term storage for computations running on the cluster
- Created automatically upon first access and visible (mounted) only when accessed
- Disk quota of 2.5/2.7 TB and a maximum of 1m/1.5m files (soft/hard quota). Quota can be checked with the command `lquota`
- Strict usage rules; see `$SCRATCH/__USAGE_RULES__` for details
- No backup

Local scratch

Local `/scratch` directory on each compute node (=`$TMPDIR`)

- Intended for serial, I/O-intensive applications
- Very short life span; data are deleted automatically when the job ends
- Scratch space must be requested by the job (see “batch system” later on)
- No backup

Data > Group storage (only shareholders) > Project vs. Work

Project

`/cluster/`**`project`**`/`*`groupname`*

- Similar to home, but for groups
- Safe, long-term storage for critical data

Work

`/cluster/`**`work`**`/`*`groupname`*

- Similar to global scratch, but without purge
- Fast, short- or medium-term storage for large computations
- Visible (mounted) only when accessed

- Shareholders can buy as much space as they need
- The access rights are managed by the owner
- Quota can be checked with `lquota`
- Backed up multiple times per week

Data > External storage > Central NAS vs. other NAS

Central NAS

Groups who have purchased storage on the central NAS of ETH can access it on our clusters

Other NAS

Groups who are operating their own NAS can export a shared file system via NFS to Euler

The user and group ID's on the NAS needs to be consistent with ETH user names and groups

- The NAS share needs to be mountable via NFSv3 (shares that only support CIFS cannot be mounted on the HPC clusters)
- The NAS share needs to be exported to the subnet of our HPC clusters.

  For central NAS, contact ID Systemdienste and ask them for an NFS export of your NAS share.
- Mounted automatically on our clusters under `/nfs/`*`servername`*`/`*`sharename`*

Data > File system comparison

File system	Life span	Snapshot	Backup	Max size	Small files	Large files
`/cluster/home`	Permanent	Yes	Yes	16 GB	+	o
`/cluster/scratch`	2 weeks	-	-	2.5 TB	o	++
`/cluster/project`	4 years	Optional	Yes	Flexible	+	+
`/cluster/work`	4 years	-	Yes	Flexible	o	++
local `/scratch`	Job	-	-	800 GB	++	o
central NAS	Flexible	Yes	Optional	Flexible	+	+

Retention time
Snapshots: up to 1 week
Backup: up to 90 days

Data > Copying data from/to the cluster (command line)

Secure copy (`scp`) is most commonly used to transfer files

```
scp [options] source destination
```

Examples: All the following examples need to be run on your local computer

- Upload a file from your workstation to Euler

```
scp local_file username@euler.ethz.ch:/path/to/remotedir
```

- Download a file from Euler to your workstation

```
scp username@euler.ethz.ch:/path/to/remote_file /path/to/localdir
```

- Copy a whole directory

```
scp -r localdir username@euler.ethz.ch:remotedir
```

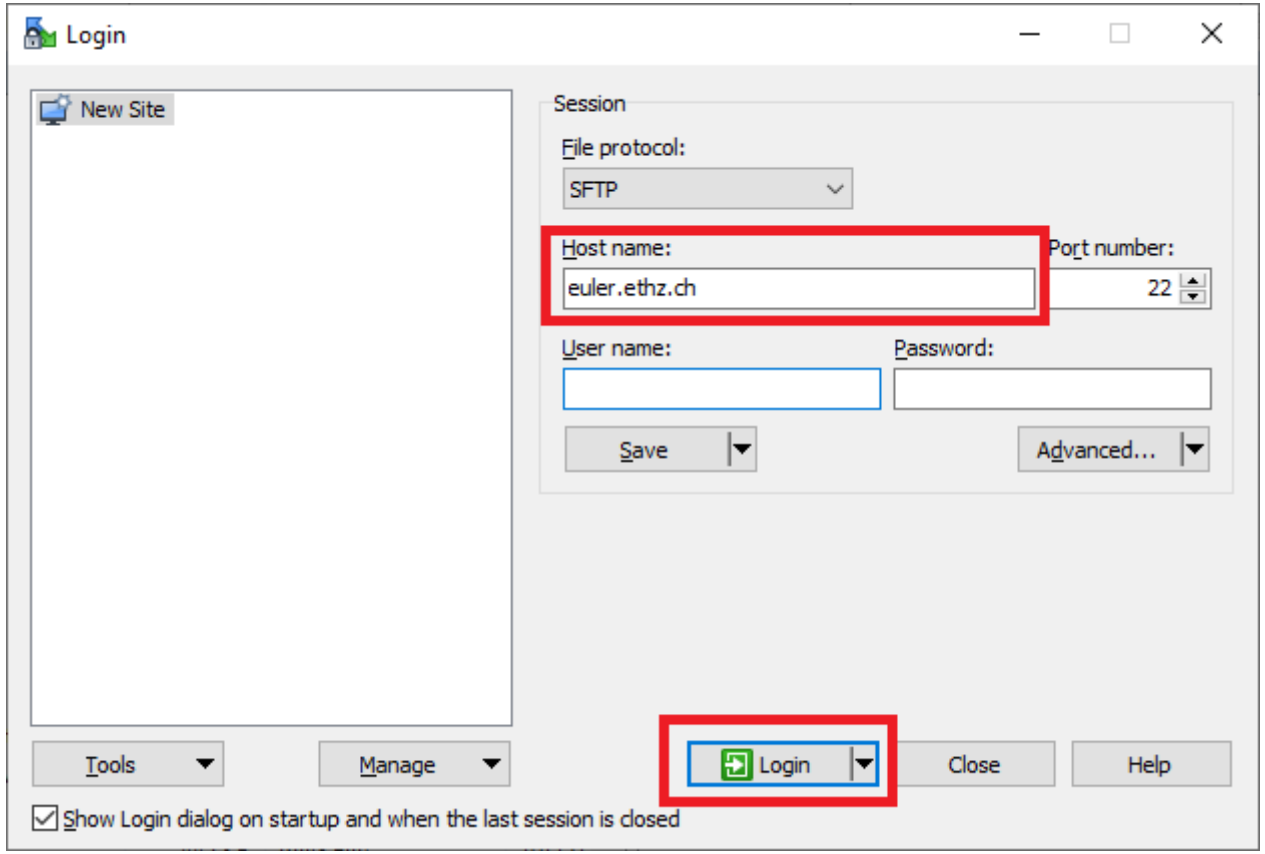
Alternatives to `scp`: `sftp`, `rsync`, `svn`, `git`, `wget`

Data > Copying data from/to the cluster (graphical user interface)

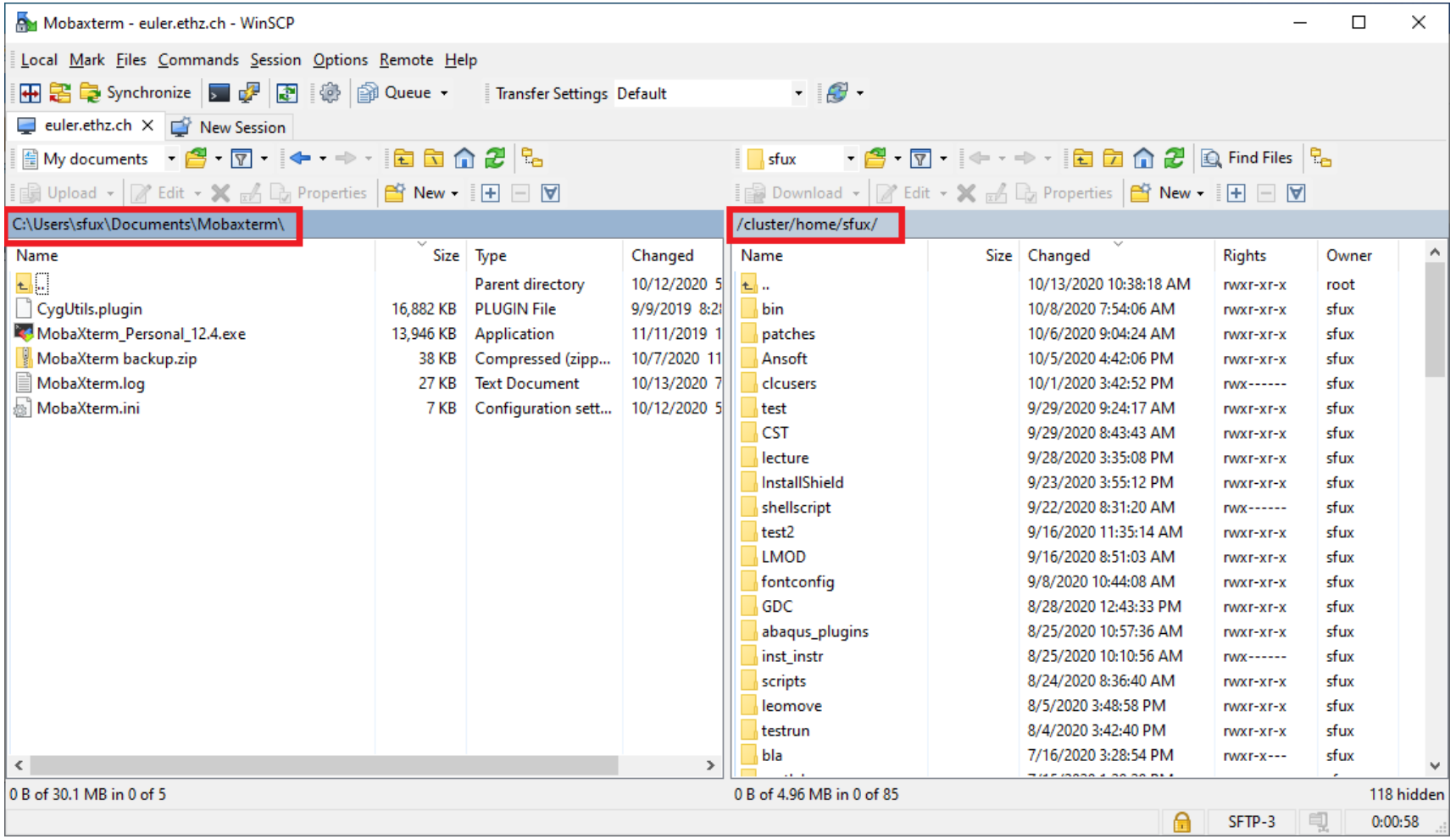
Graphical file transfer programs

Linux	macOS	Windows
FileZilla	FileZilla Cyberduck	WinSCP PSCP FileZilla Cyberduck

Data > Copying data from/to the cluster > WinSCP



Data > Copying data from/to the cluster > WinSCP



Data > Exercise 1: Download data

Tasks	Commands
Log in to Euler	`ssh username@euler.ethz.ch`
Check your quota	`lquota`
Go to your home folder	`cd $HOME`
Download the example repository with git	`git clone https://gitlab.ethz.ch/jarunanp/hpc-examples.git`
Go to python multiprocessing example directory	`cd hpc-examples/python/multiprocessing`
Switch to the new software stack	`env2lmod`
Load modules	`module load gcc/6.3.0 python/3.8.5`
Request an interactive session with a compute node	`srun --ntasks=2 --pty bash`
Run the python script, output result to a logfile (*n=1,2*)	`python process.py` *n* `> process_`*n*`.log`
Exit the interactive session	`exit`

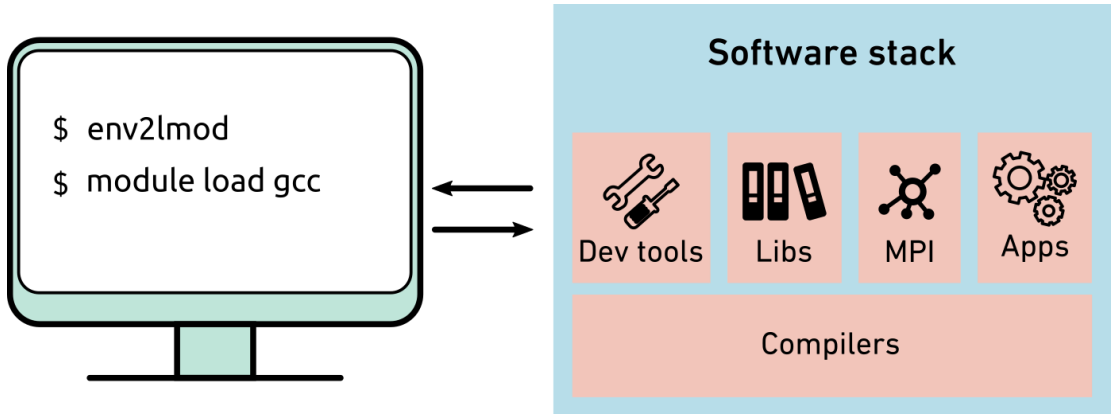
Data > Exercise 2: Data transfer

Tasks	Commands
On your local computer, copy the logfiles from Euler	`scp username@euler.ethz.ch:/cluster/username/home/hpc-examples/python/multiprocessing/process*.log .`
	`rsync -av username@euler.ethz.ch:/cluster/username/home/hpc-examples/python/multiprocessing/process*.log .`
Copy the whole folder with `scp -r`	`scp -r username@euler.ethz.ch::/cluster/username/home/hpc-examples/python/multiprocessing .`
Try `rsync`	`rsync -av username@euler.ethz.ch:/cluster/username/home/hpc-examples/python/multiprocessing .`
	`rsync -av username@euler.ethz.ch:/cluster/username/home/hpc-examples/python/multiprocessing ./`
	`rsync -av username@euler.ethz.ch:/cluster/username/home/hpc-examples/python/multiprocessing/* ./`

Outlook

- Introduction
- Accessing the cluster
- Storage and data transfer
- Modules and applications

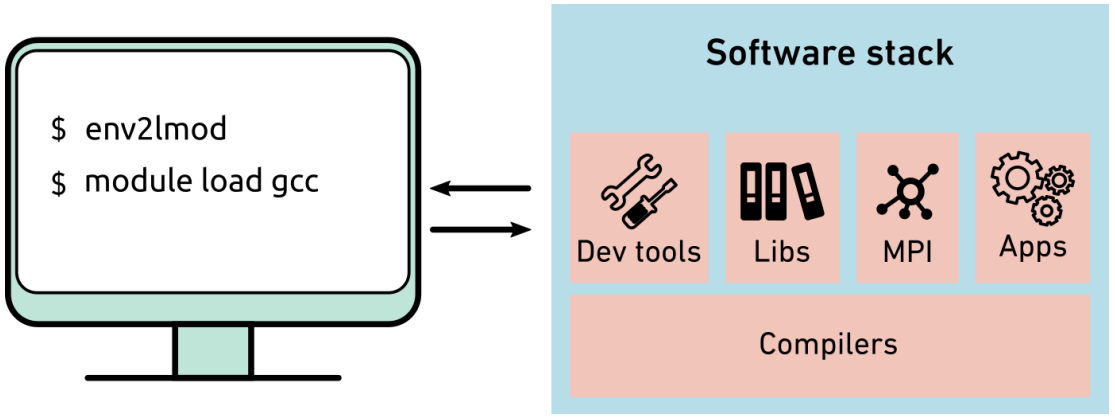
Modules



A Modules package is a tool to let users easily configure the computing environment which includes

- Development tools
- Scientific libraries
- Communication libraries (MPI)
- Third-party applications

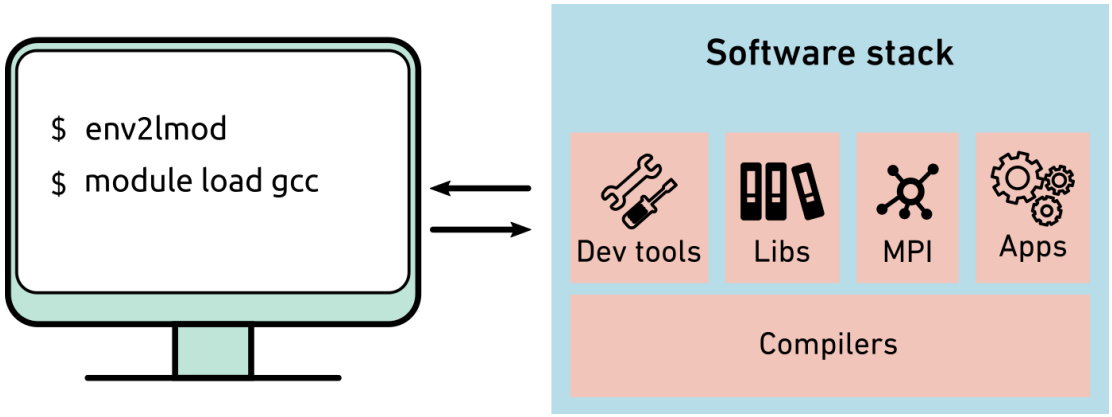
Modules



Advantages:

- Automatic configuration
- Different versions of the same software can co-exist and can be selected explicitly
- You can easily try out different tools, switch between versions, to find out which one works best for you

Modules



We employ two Modules packages on the cluster

- **LMOD Modules**
- **Environment Modules**

Modules > Commands

Load Python module in GCC/6.3.0 toolchain

```
[sfux@eu-login-31 ~]$ module load gcc/6.3.0 python/3.8.5
```

List available python module

```
[sfux@eu-login-31 ~]$ module avail python
-------------------- /cluster/apps/lmodules/Compiler/gcc/6.3.0 ---------------------
python/2.7.14    python/3.6.4    python/3.7.4    python/3.8.5 (D)    python_gpu/3.8.5
```

List all currently loaded modules

```
[sfux@eu-login-31 ~]$ module list

Currently Loaded Modules:
  1) StdEnv   2) gcc/6.3.0   3) openblas/0.2.20   4) python/3.8.5
```

Modules > Commands

`module`	get info about module sub-commands
`module avail`	list all modules available on the cluster
`module key` *`keyword`*	list all modules whose description contains *keyword*
`module help` *`name`*	get information about module *name*
`module show` *`name`*	show what module *name* does (without loading it)
`module unload` *`name`*	unload module *name*
`module purge`	unload all modules at once

Modules > Where are applications installed?

Centrally installed in /cluster/apps

Applications that are needed by many users should be installed centrally, like compilers and libraries

- Visible and accessible to all users via modules
- Installed and maintained by cluster support
- Commercial licenses provided by the IT shop of ETH or by research groups

In $HOME

Users can install additional applications in their home directory, but only if the quotas (space: 16 GB, files/directories: 200'000) are not exceeded

- Avoid anaconda installations as they often conflict with the files/directories quota. Alternatively, you can create a Python virtual environment.
- For Python and R, packages can easily be installed locally

```
$ pip install --user packagename
```

Modules > Two software stacks

Old software stack with Env Modules

- The old software stack is set as a default.
- The old software stack is provided for reproducibility

To switch from the old to the new software stack

```
$ env2lmod
```

To set the default software stack to the new one

```
$ set_software_stack.sh new
```

New software stack with LMOD Modules

- All new software is installed exclusively in the new software stack

To switch from the new to the old software stack

```
$ lmod2env
```

https://scicomp.ethz.ch/wiki/Setting_permanent_default_for_software_stack_upon_login

Modules > Two software stacks

Old software stack with Env Modules

Applications and tools were installed manually

- Mostly compiled with the standard/system compiler
- Limited implementation of toolchains available
- Only most important libraries available for all compilers/MPI
- Three module categories (legacy/supported/new)

New software stack with LMOD Modules

Installations mostly done with SPACK package manager

- Automatic compilation/installation directly from source code
- Automatic creation of module files (LMOD)
- Predefined build recipes for more than 6’700 applications and libraries
- Large community behind SPACK (LLNL, ORNL, NASA, ETH, EPFL etc.)

https://scicomp.ethz.ch/wiki/New_SPACK_software_stack_on_Euler

Modules > Two software stacks

Old software stack with Env Modules

Application pages (for instance R, Python, Matlab etc.) containing installation guides and examples how to submit batch jobs are also valid for the new software stack.

https://scicomp.ethz.ch/wiki/Category:Application

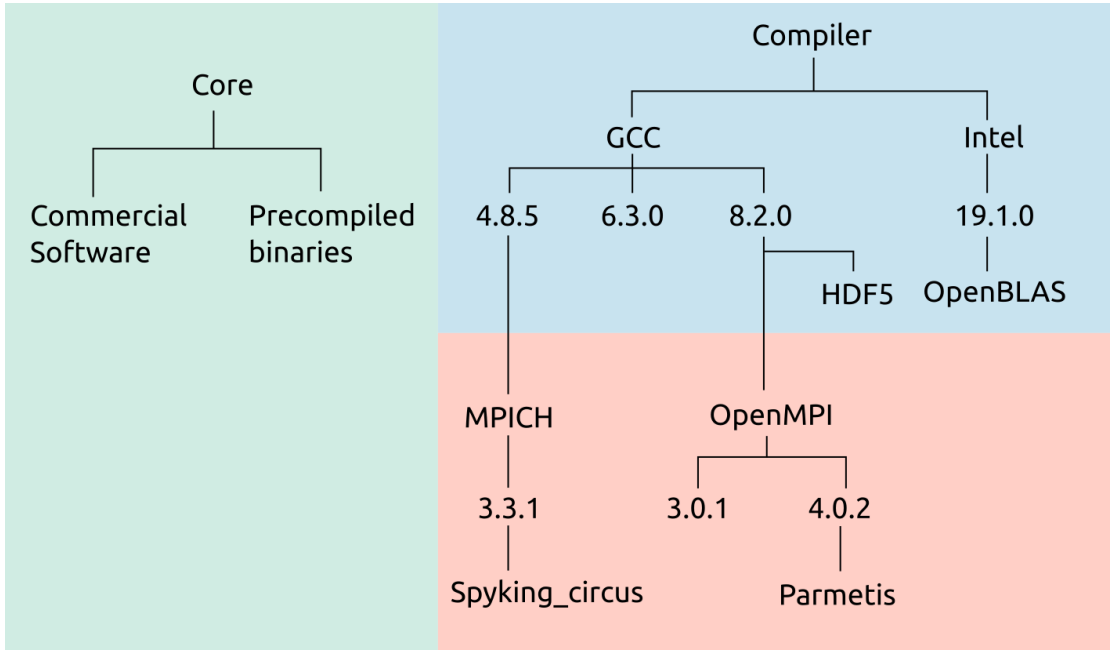
New software stack with LMOD Modules

Application lists:

https://scicomp.ethz.ch/wiki/Euler_applications_and_libraries

https://scicomp.ethz.ch/wiki/Python_on_Euler

Modules > LMOD modules



Module hierarchy with 3 layers:

- Core layer

```
$ module load comsol/5.6
```

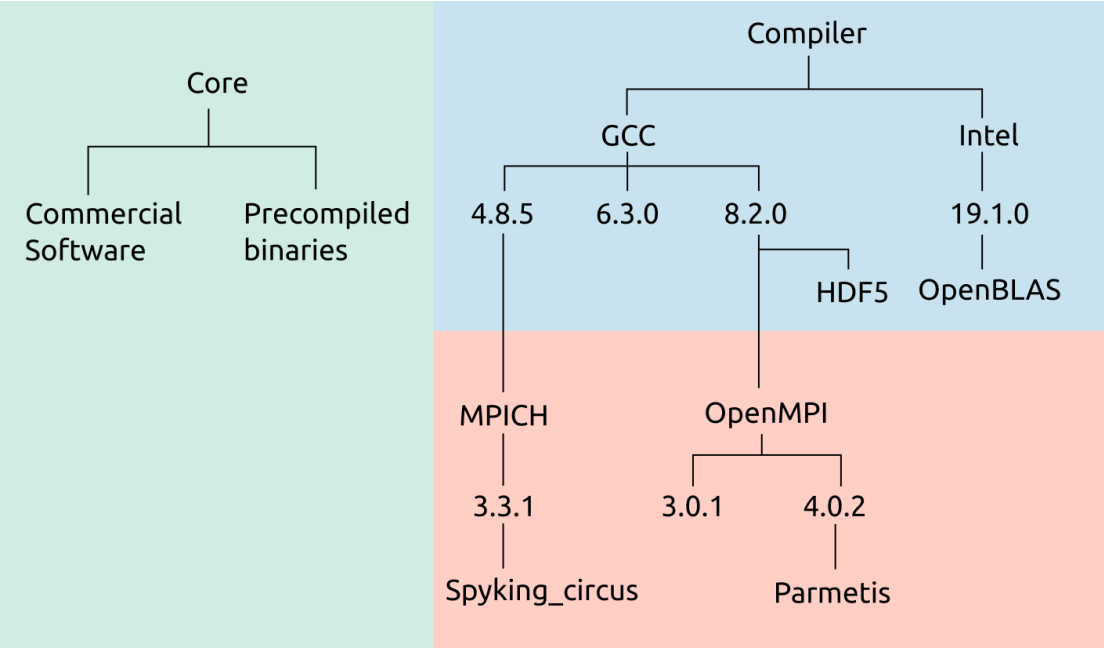
- Compiler layer

```
$ module load gcc/6.3.0 python/3.8.5
```

- MPI layer

```
$ module load gcc/6.3.0 openmpi/4.0.2 openblas
```

Modules > LMOD modules

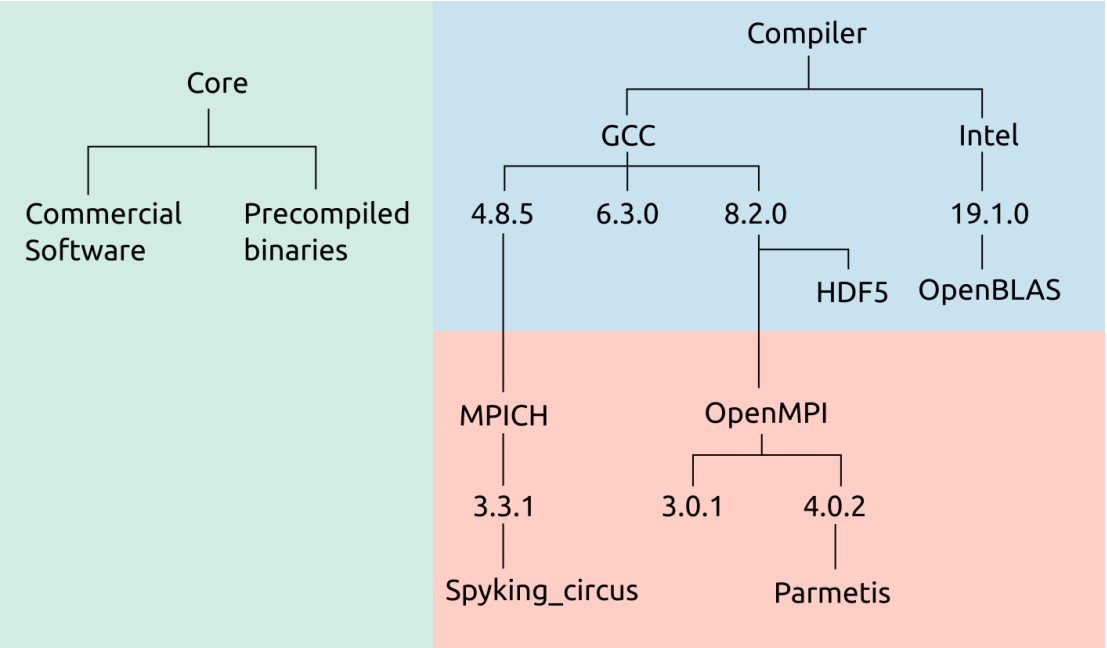


The four main toolchains are

1. GCC 4.8.5 (supports C++11 standard)
2. GCC 6.3.0 (supports C++14 standard)
3. GCC 8.2.0 (supports C++17 standard)
4. Intel 19.1.0

- Combined with OpenMPI 3.0.1, 4.0.2 or 4.1.4 and OpenBLAS
- For each of the toolchains ~400-500 applications and libraries are available, including many Linux low level libraries

Modules > LMOD modules



- Safety rules to avoid misconfiguration
- Only one compiler/MPI combination can be loaded at the same time
- When changing the compiler or MPI, LMOD will try to reload all currently loaded modules with the new compiler/MPI

Modules > Commercial / Open source applications

Categories	Application lists
Bioinformatics and life sciences	Bamtools, BLAST, Bowtie, RAxML, Relion, TopHat
Finite element methods	Ansys, Abaqus, FEniCS
Machine learning	PyTorch, Scikit-Learn, TensorFlow, Theano
Multi-physics phenomena	Ansys EDT, COMSOL Multiphysics, STAR-CCM+
Quantum chemistry and molecular dynamics	ADF, Ambertools, Gaussian, Molcas, Octopus, Orca, Qbox, Turbomole
Symbolic, numerical and statistical mathematics	Gurobi, Mathematica, MATLAB, R, Stata
Visualization	Ffmpeg, ParaView, VisIT, VTK

Modules > Software development

Development tools	
Compilers	GCC, Intel
Programming languages	C, C++, Fortran, Go, Java, Julia, Perl, Python, Ruby, Scala
Scientific libraries	Boost, Eigen, FFTW, GMP, GSL, HDF5, MKL, NetCDF, NumPy, OpenBLAS, SciPy
Solvers	PETSc, Gurobi, Hypre, Trilinos
MPI libraries	Open MPI, Intel MPI, MPICH
Build systems	GNU Autotools, Cmake, qmake, make
Version Control	CVS, Git, Mercurial, SVN

Modules > Exercise 1

Tasks	Commands
Go to the mpi4py in the example folder	`cd $SCRATCH/hpc-examples/python/mpi4py`
Switch to the new software stack	`env2lmod`
Or, set the new software stack to default	`set_software_stack.sh new`
Switch to the GCC 6.3.0 toolchain	`module load gcc/6.3.0`
Check available openmpi and python versions	`module avail openmpi` `module avail python`
Load new openmpi and python modules	`module load openmpi/4.0.2 python/3.8.5`
Submit a job	`sbatch --ntasks=4 --time=5 --mem-per-cpu=500 --wrap="mpirun python hello_mpi4py.py"`
Check job status	`squeue`

Modules > Exercise 2: Create a Python environment

Tasks	Commands
Go to your scratch folder	`cd $SCRATCH`
Switch to the new software stack	`env2lmod`
Or, set the new software stack to default	`set_software_stack.sh new`
Load modules	`module load gcc/6.3.0 python/3.8.5`
Check pip version	`pip --version`
Create a virtual environment	`python -m venv --system-site-packages demo_env`
Activate the environment	`source demo_env/bin/activate`
Now you should see that your environment is enabled as your prompt started with (`demo_env`)	-
Check pip version	`pip --version`
See all python packages	`pip list`
Upgrade pip	`pip install --upgrade pip`
Check the pip version again	`pip --version`

Questions?