VSC HPC Introduction

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

Vlaanderen
is supercomputing

VLAAMS
SUPERCOMPUTER
CENTRUM | *Innovative Computing for A Smarter Flanders*

vscentrum.be

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 …

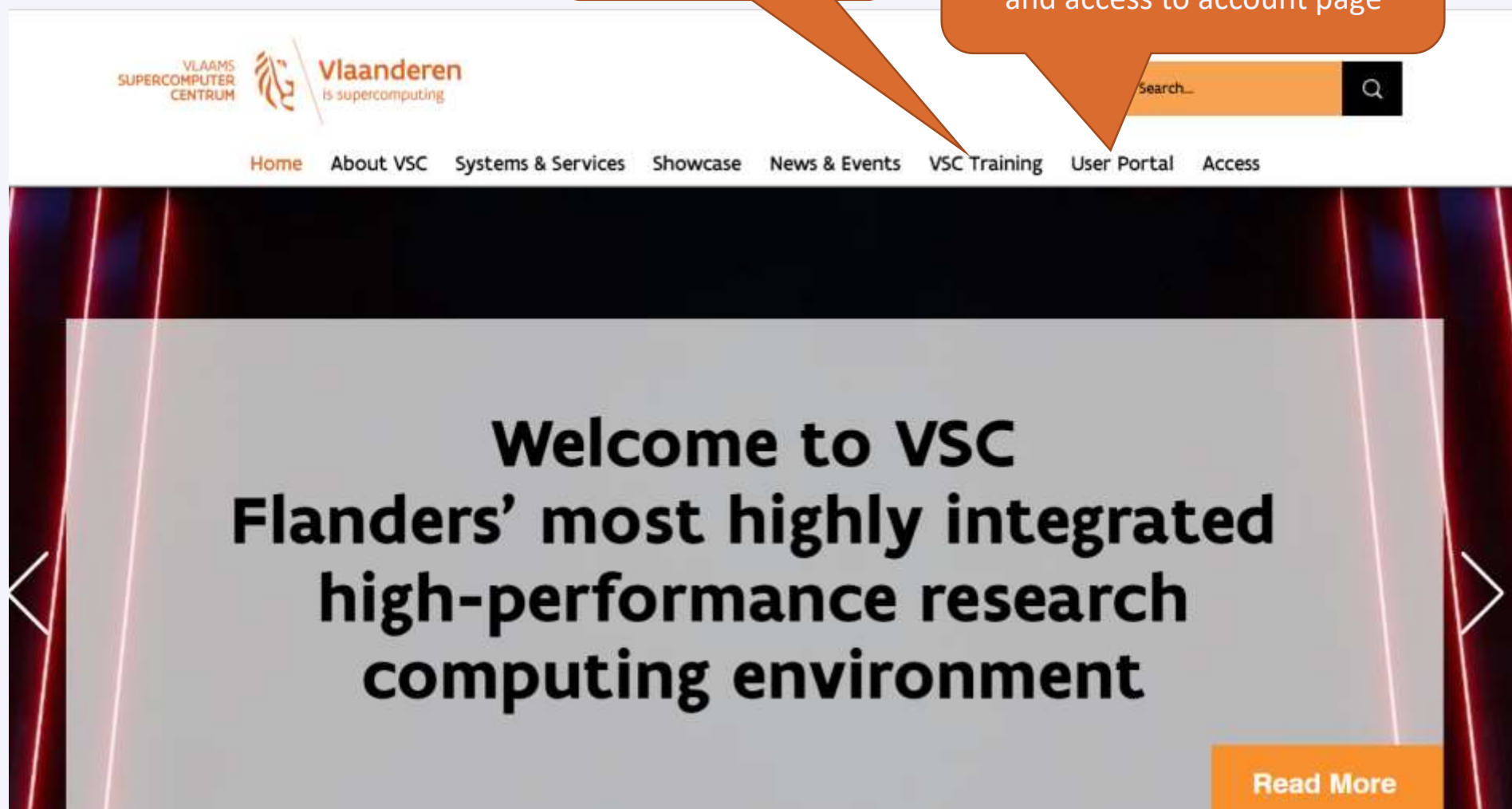The concept is simple: **Parallelism** = employing multiple processors for a single problem

VLAAMS
SUPERCOMPUTER
CENTRUM

To manage your VSC account:
account.vscentrum.be

Upload more SSH public keys

Upload SSH Key Request account

Create or request to join a new group

Moderate your own groups (add/remove users)

account.vscentrum.be/django/

View account

General information

Uid:
Institute: Leuven / Hasselt
Institute login:
Gecos:

View Account | Edit Account | View Groups | New/Join Group | Edit Group | New/Join VO | Reservations | Log Out

VLAAMS SUPERCOMPUTER CENTRUM

Support and Services

Basic support
- Helpdesk (**hpcinfo@kuleuven.be**)
- Monitoring and reporting

Application support
- Installation and porting
- Optimisation and debugging
- Benchmarking
- Workflows and best practices

Training
- Documentation and tutorials
- Scheduled trainings / workshops
- On request workshops
- One-to-one sessions

VLAAMS
SUPERCOMPUTER
CENTRUM

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
- ❑ All service costs (compute and storage) are all explained
  Go to ICTS service catalogus: https://icts.kuleuven.be/sc
  Click on High Performance Computing (NL/EN)

VLAAMS
SUPERCOMPUTER
CENTRUM
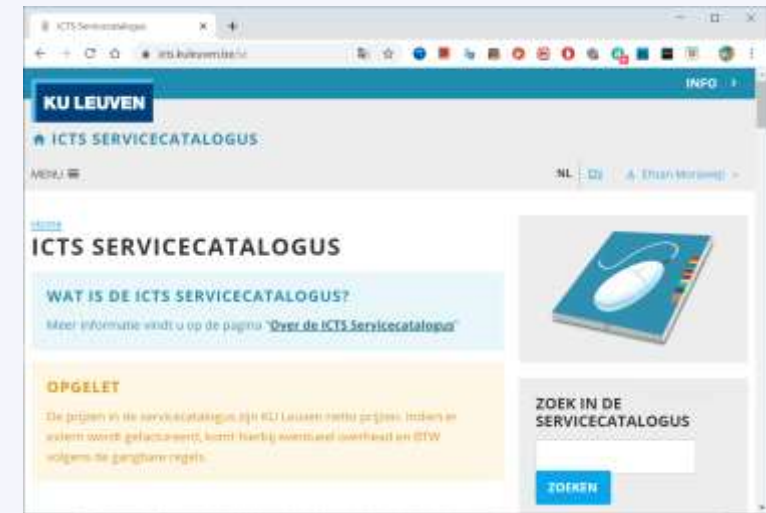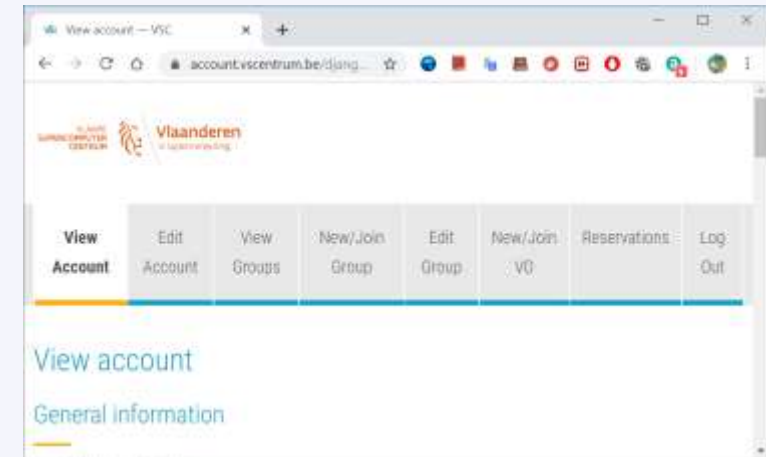
VSC training

- Introductory

  Matlab (Supercalculator)
  Matlab Programming

  Linux → HPC intro

  Linux for HPC → Linux scripting → Linux tools

  Make intro

  CMake intro

  Version control systems

  C

- Intermediate

  C++ for scientific computing

  Fortran for programmers

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

  worker/atools

  Julia good bad ugly

- Advanced

  High Performance Python

- Specialized track

  ?

  MPI

  OpenMP

  Debugging techniques

  Code optimization

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

VLAAMS
SUPERCOMPUTER
CENTRUM

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.

VLAAMS
SUPERCOMPUTER
CENTRUM

Tier-2 Clusters

VSC HPC Environments

Tier-0

PRACE

Vlaanderen
is supercomputing

Tier-1

Hortense

Tier-2

GHENT UNIVERSITY

VUB VRIJE UNIVERSITEIT BRUSSEL

Universiteit Antwerpen

KU LEUVEN ▶▶ UHASSELT

STEVIN   **HYDRA**   **HOPPER/LEIBNIZ**   **GENIUS/WICE**

VLAAMS
**SUPERCOMPUTER
CENTRUM**

Tier-2 Clusters @ KU Leuven

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

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

VLAAMS
SUPERCOMPUTER
CENTRUM

Tier-2 Cluster - Genius

Node type	CPU type	# cores	Total mem	Mem per core (MB)	Partition
Cascadelake	Xeon Gold 6240	36	**192 GB**	**5000**	batch
Skylake **large mem**	Xeon Gold 6240	36	**768 GB**	**21000**	bigmem
Skylake **GPU**	Xeon Gold 6140 **4xP100** SXM2 16GB	36	**192 GB**	**5000**	gpu_p100
Cascadelake **GPU**	Xeon Gold 6240 **8xV100** SXM2 32GB	36	**768 GB**	**21000**	gpu_v100
Superdome	Xeon Gold 6132 8 sockets	112	**6 TB**	**53500**	superdome

○ **End of life:** end 2024   ○ **SSD Disks:** 200 GB

Remarks

○ **Access:** login.hpc.kuleuven.be

○ **Interconnect:** InfiniBand EDR (25 Gb/s)

VLAAMS
SUPERCOMPUTER
CENTRUM

Tier-2 Cluster - wICE

Node type	CPU type	# cores	Total mem	Mem per core (MB)	Partition
Icelake	Xeon 8360Y	72	256 GB	3455	batch_icelake
Sapphire Rapids	Xeon 8468	96	256 GB	2500	batch_sapphirerapids
Icelake **large mem**	Xeon 8360Y	72	2 TB	28000	bigmem
Icelake **large mem**	Xeon 8360Y	72	8 TB	111900	hugemem
Icelake **GPU**	Xeon 8360Y **4xA100** SXM2 80GB	72	512 GB	7000	gpu_a100
AMD **GPU**	Epyc 9334 **4xH100** SXM5 80GB	64	768 GB	11700	gpu_h100
Icelake **Interactive**	Xeon 8358 **1xA100** SXM2 80GB	64	512 GB	7500	interactive

Remarks

- **Phase 1:** since September 2022
- **Access:** login.hpc.kuleuven.be
- **Interconnect:** InfiniBand HDR (100 Gb/s)

- **SSD Disks:** 960 GB
- **Interactive partition:**
  max resources: 8 cores, 1 GPU, 16hr

VLAAMS SUPERCOMPUTER CENTRUM

Tier-2 Cluster - wICE

Compute nodes
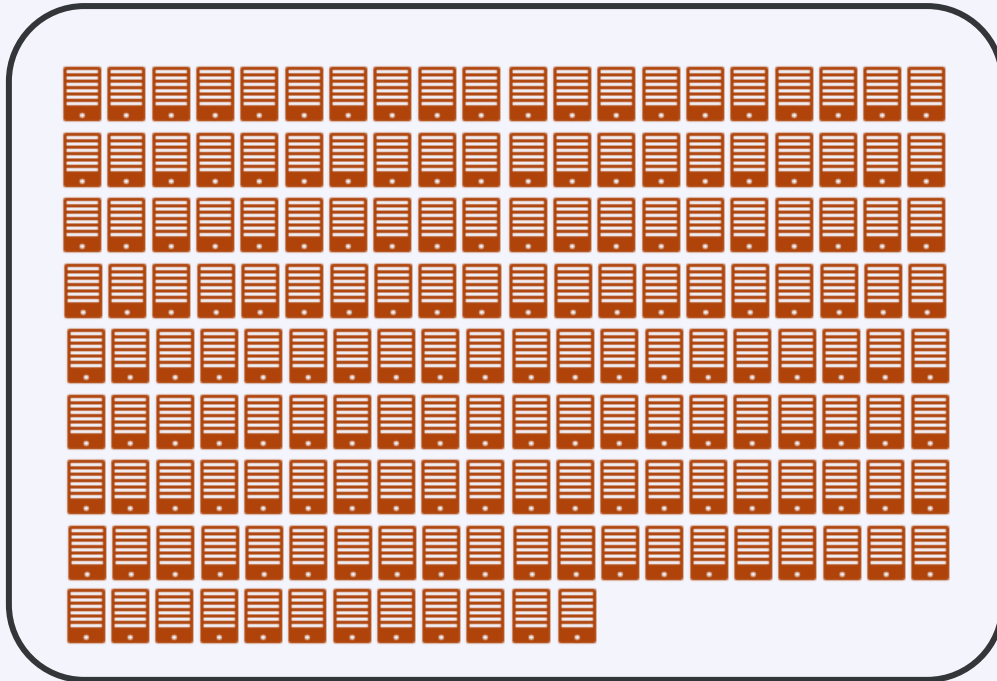
Large memory nodes

GPU nodes

5x IceLake 72c 2048 GB

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

Interactive nodes

No Dedicated Login Nodes

172x IceLake 72c 256 GB
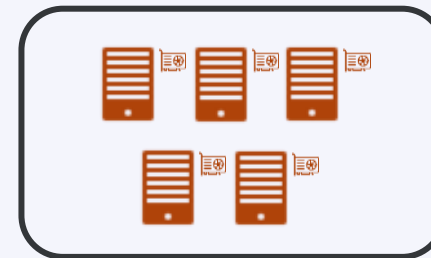
4x IceLake 64c 512 GB
1 A100 80GB

VLAAMS
SUPERCOMPUTER
CENTRUM

Tier-2 Cluster – wICE extension

Compute nodes

Large memory nodes
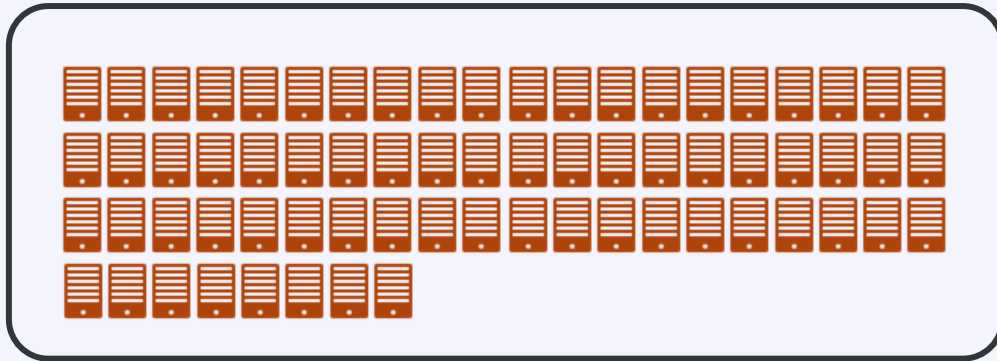
GPU nodes

68x Sapphire Rapids 96c 256 GB
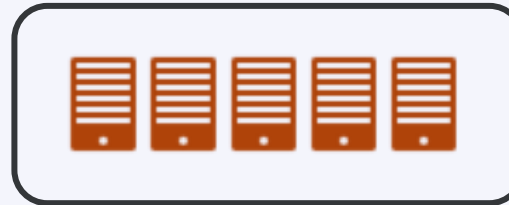
1x IceLake 72c 8 TB

4x AMD Genoa 64c 768 GB
4x H100 SXM5 80GB

VLAAMS
SUPERCOMPUTER
CENTRUM

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)

VLAAMS
SUPERCOMPUTER
CENTRUM

Storage

- **Do NOT use `/tmp`**
  It is only 10 GB and is reserved for the OS and root processes.
  Your application can crash if using `/tmp`

- You are automatically logged into your home folder upon login.
  Immediately go to your other storages, e.g.
  `$ cd $VSC_DATA`

- Always check your storage balance using `myquota` command

```
$ 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

VLAAMS
SUPERCOMPUTER
CENTRUM
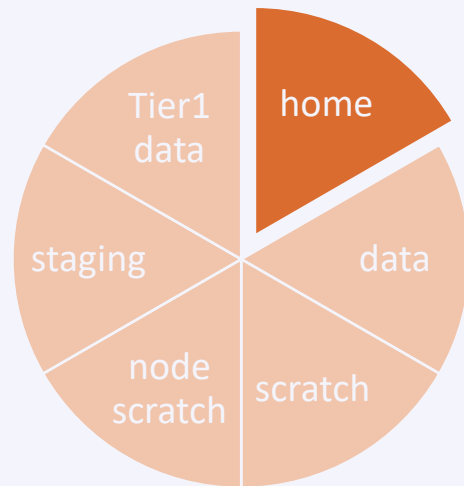
Storage

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

VLAAMS
SUPERCOMPUTER
CENTRUM
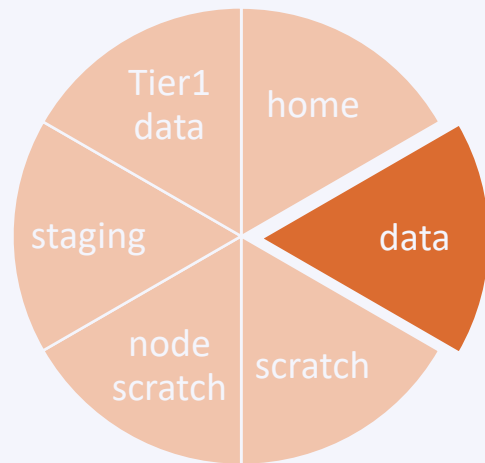
Storage

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

VLAAMS
SUPERCOMPUTER
CENTRUM
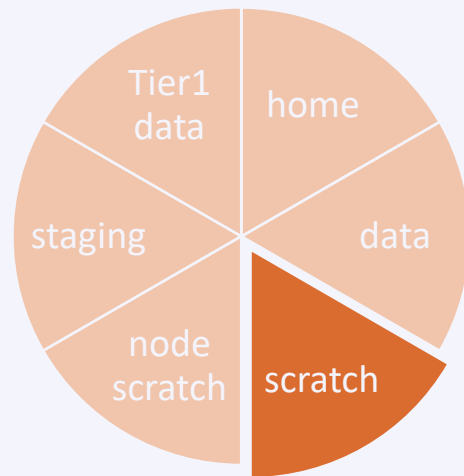
Storage

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	- Recommended storage for all jobs - Copy scratch files to VSC_DATA or local storage after jobs are done - Deleted files cannot be recovered

VLAAMS
SUPERCOMPUTER
CENTRUM
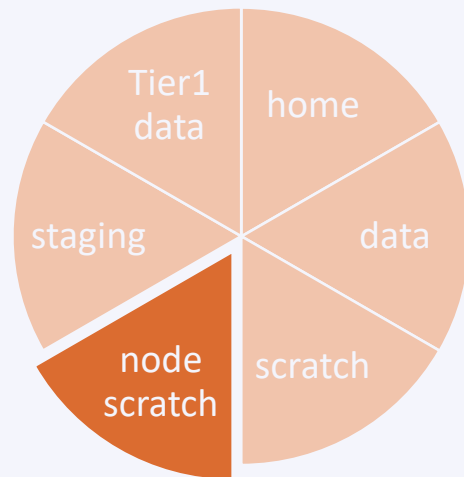
Storage

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

VLAAMS
SUPERCOMPUTER
CENTRUM

Login Nodes & MFA

Accessing a Cluster

Compute nodes

Firewall

Infiniband interconnect

Link to other VSC sites

Scientist

MFA or SSH Authentication

Login nodes

Storage

Service nodes

Administration - queue system, job scheduler, user management,…

VLAAMS SUPERCOMPUTER CENTRUM
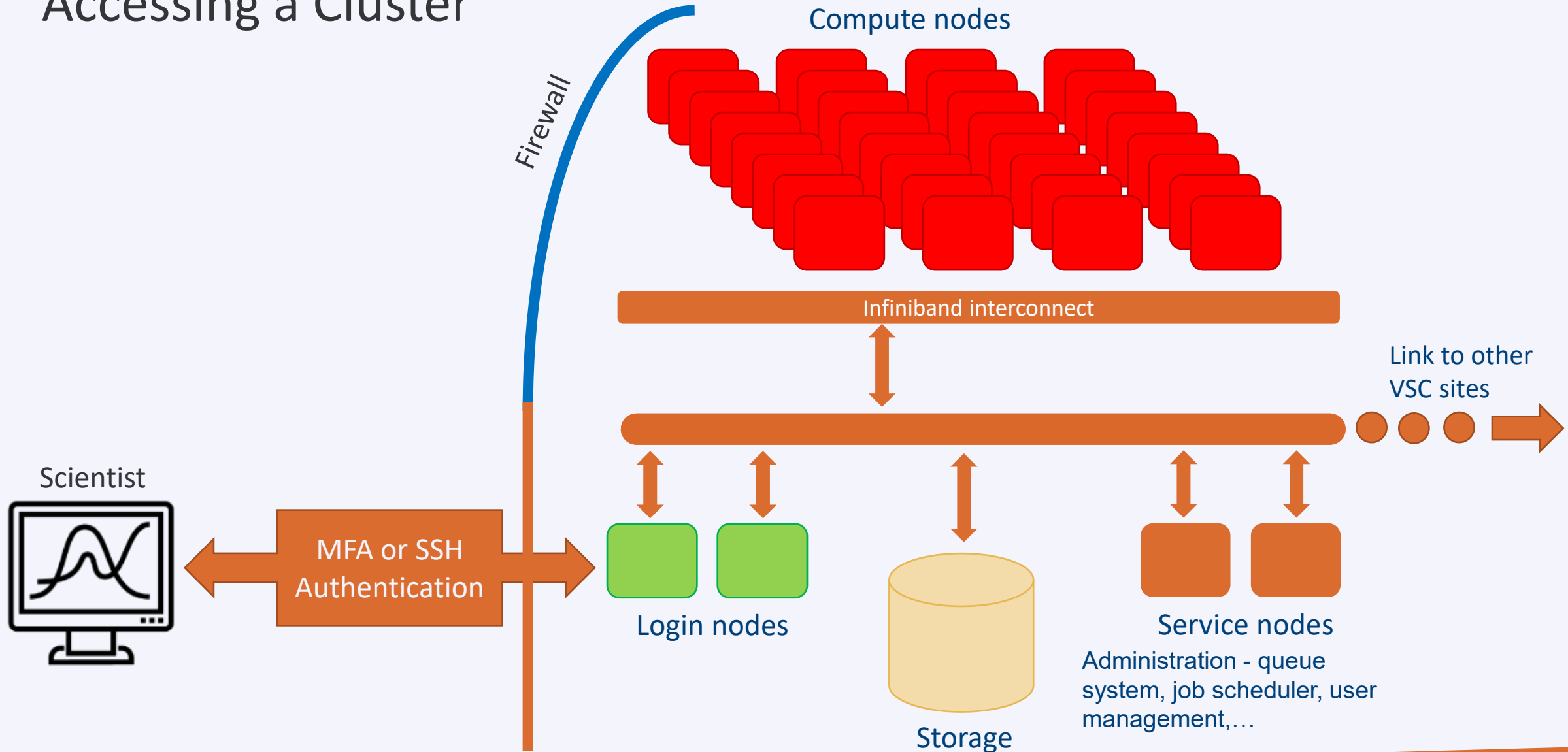
How to login?

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

NX

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

SSH client

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

VLAAMS
SUPERCOMPUTER
CENTRUM

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)
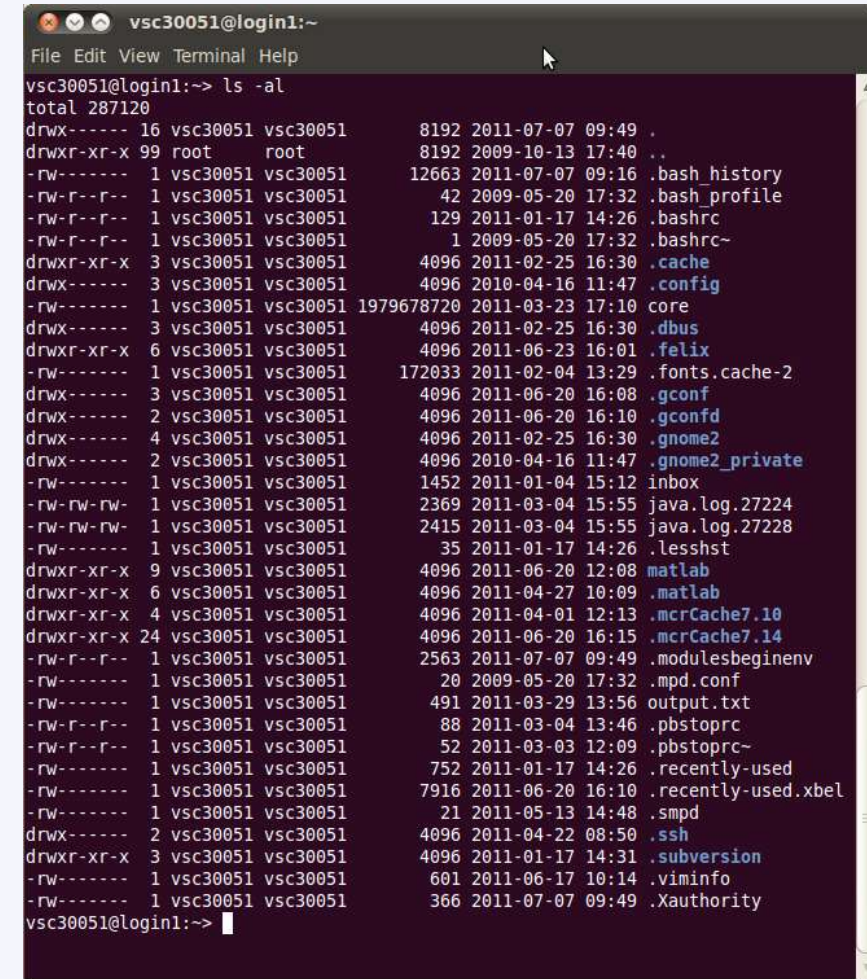
Login nodes are shared resources

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

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

Warning

VLAAMS
SUPERCOMPUTER
CENTRUM

Command Line

- Text mode
- Provides a way to control the computer via the keyboard.
- Is common use on HPC
- Is somewhat archaic, but very powerful.
- Can be much quicker than a GUI.
- Implicitly assumes that you know what your are doing
- Don't be scared!
- If anything goes wrong, you can stop the command with **Ctrl+C**

VLAAMS
**SUPERCOMPUTER
CENTRUM**

Important rules on the command line

1. Linux systems are case (and space) sensitive.
   - `MyFile` is not same as `myfile`
2. There is no "recycle bin" or "trash can" when working in the command line environment. There might be one for GUI.
   When files are deleted on the command line, they instantly disappear forever.
3. You should always practice new commands on a test system that is not used in a production environment. This minimizes the chances of an accident that can take down an important system

Important rules on the command line

- "\" vs. "/":
  - In Linux, the "/" is the directory separator, and the "\" is an escape character.
  - In Windows, the forward-slash "/" is the command argument delimiter, while the backslash "\" is a directory separator

- Filenames:
  - In Linux, there is no such thing as a file extension.
    Periods can be placed at any part of the filename, and "extensions" may be interpreted differently by all programs, or not at all.
  - Windows uses the ".extension" filename convention, (e.g. FILENAME.TXT).

Getting help: command built-in

- Help on most Linux commands is typically built into the command themself
- These flags usually look like "`-h`" or "`--help`".
- `$ ls --help`
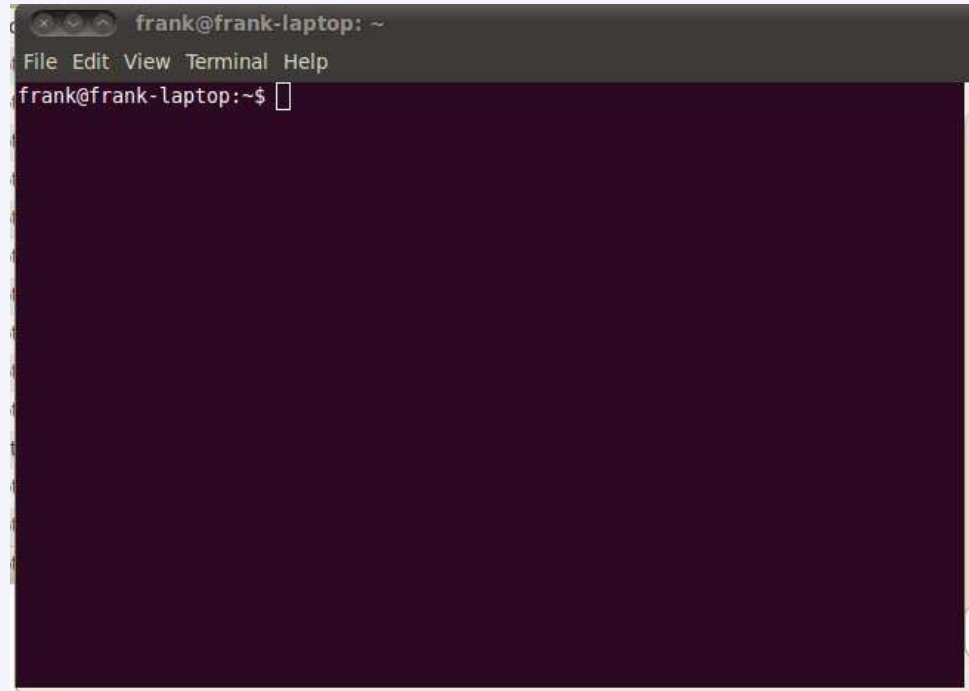
Getting help: man pages

- Best source of information can be found in the online manual pages, "**man pages**" for short.
  type "`man command`".

  ```
  $ man grep
  ```

- Tips:
  - To search for a particular word (e.g. file) within a man page, type "`/word`".
  - To quit from a man page, type the "`q`" key.
  - If you do not remember the name of Linux command and you know a keyword relating to the command, search the man pages with the `-k`
    ```
    $ man -k control
    ```

Getting help: info pages

- Info pages are similar to man page, but instead of being displayed on one long scrolling screen, they are presented in shorter segments with links to other pieces of information.

- Access with the "`info`" command
  `$ info ls`

- Tips:
  - To quit from a info page, type the "`q`" key.
  - Type "h" to get more help on the info

VLAAMS
SUPERCOMPUTER
CENTRUM

Getting help

- bash has a built-in help facility available for each of the shell *builtins*.
  ```
  $ help cd
  $ help pwd
  ```

- `whatis` displays a very brief description of a command
  ```
  $ whatis pwd
  ```

VLAAMS
SUPERCOMPUTER
CENTRUM

Useful commands

- Clear the contents of the current screen
  `$ clear`
- `$ logout`
  - The logout command logs your account out of the system (in a text mode).
  - This will end your terminal session and return to the login screen.
  - Some systems may have a file called `.logout` or `.bash_logout` in each user's home directory.
- `$ exit`
  - Exit the current shell. The `exit` command is similar to the `logout` command with the exception that it does not run the logout script located in the user's home directory.

VLAAMS
SUPERCOMPUTER
CENTRUM

Linux File System

- *hierarchical directory structure*: files are organized in a tree-like pattern of directories (folders), which may contain files and other directories, etc.
- Everything is a file:
  - Regular files
  - Directories: files listing a set of files
  - Symbolic links: files referring to the name of another file
- *root /: th*e first directory in the file system.
- Note: comparison with Windows,
  - Windows has a separate file system tree for each storage device (e.g. C-drive, D-drive, I-drive, …)
  - Linux has a single file system tree, regardless of how many drives or storage devices are attached to the computer.
    Storage devices are attached (or *mounted*) at various points on the tree.

Linux File System

Source: http://linuxsuperuser07.blogspot.be/2011/09/rhel-6-file-system.html
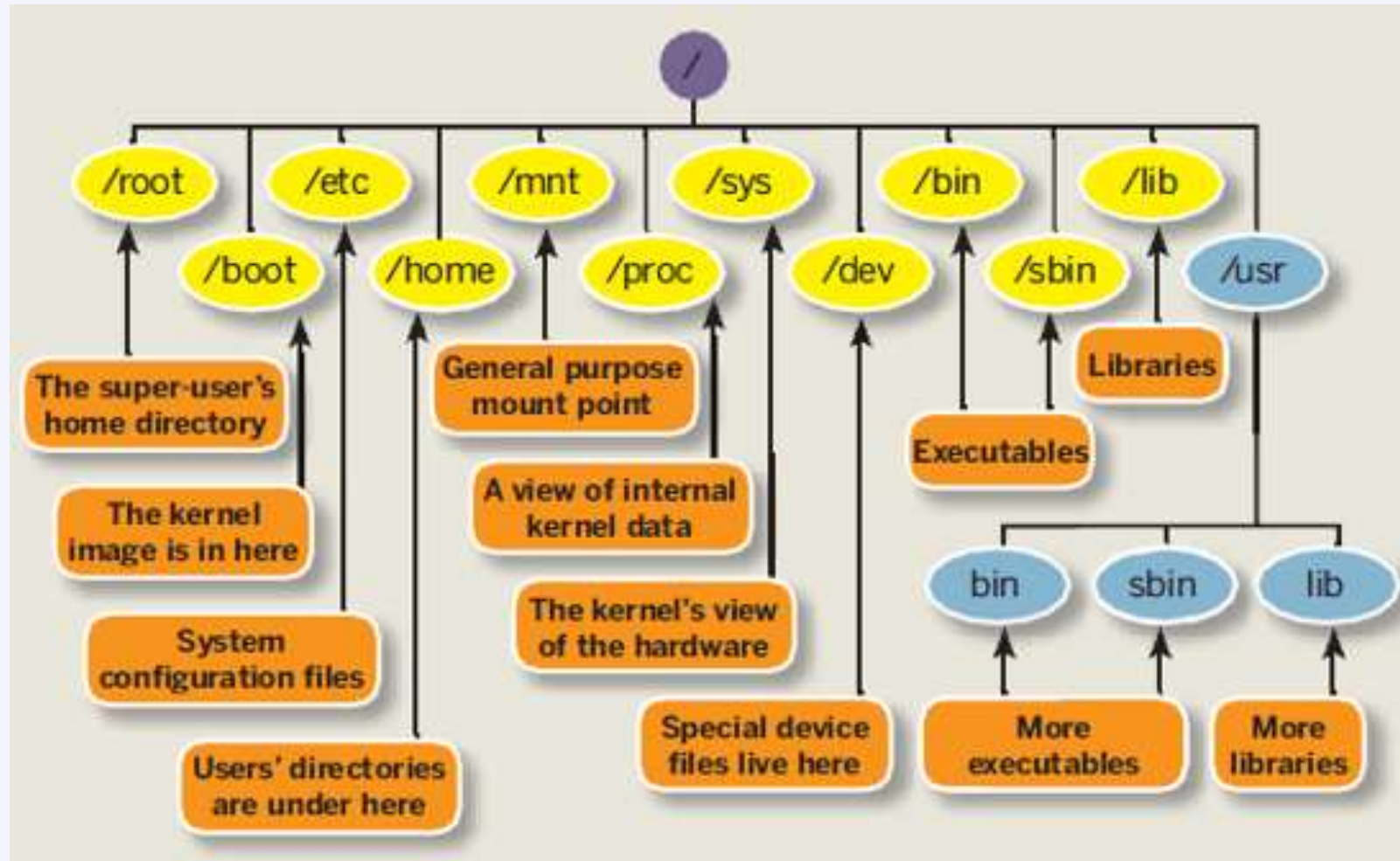
Linux File System

Not imposed by the system. Can vary from one system to the other, even between two GNU/Linux installations!

```
/               Root directory
/bin/           Basic, essential system commands
/boot/          Kernel images, initrd, configuration files
/dev/           Files representing devices
/etc/           System configuration files
/home/          User directories
/lib/           Basic system shared libraries

/media/             Mount points for removable media
```

Linux File System

```
/lost+found/    Corrupt files the system tried to                    recover
/mnt/           Mount points for temporarily mounted                 filesystems
/opt/           Specific tools installed by the                      sysadmin
/usr/local/  often used                          instead
/proc/          Access to system information
/sbin/          Administrator-only commands
/sys/           System and device controls
/tmp/           Temporary files
```

The Unix filesystem structure is defined by the Filesystem Hierarchy Standard (FHS):
http://www.pathname.com/fhs/pub/fhs-2.3.html

http://en.wikipedia.org/wiki/Filesystem_Hierarchy_Standard

Linux File System-home directory

Source: http://www.linuxplanet.com/linuxplanet/tutorials/6666/1/screenshot3894/
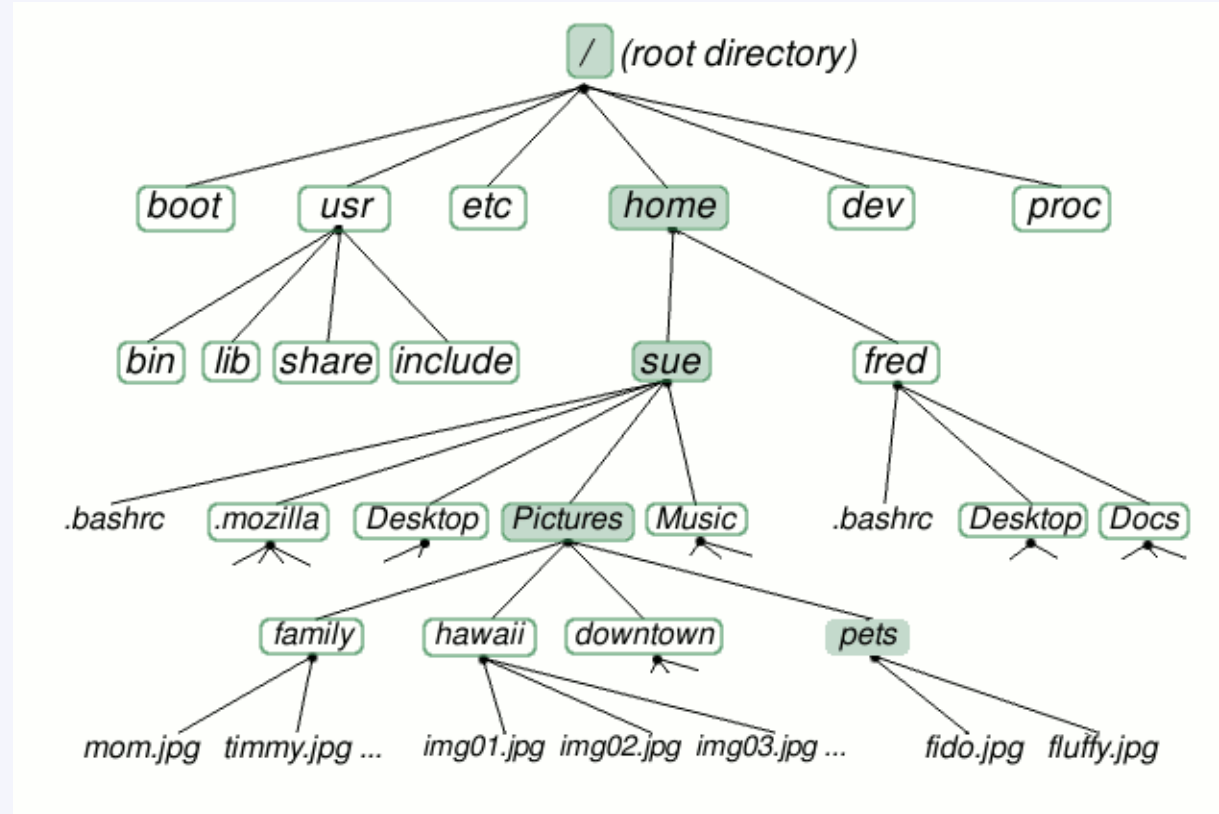
ls command

Lists the files in the current directory, in alphanumeric order, except files starting with the "." character.

- `$ ls -a` (all)
  Lists all the files (including .* files)

- `$ ls -l` (long)
  Long listing (type, date, size, owner, permissions)

- `$ ls -t` (time)
  Lists the most recent files first

- `$ ls -S` (size)
  Lists the biggest files first

- `$ ls -r` (reverse)
  Reverses the sort order

- `$ ls -ltr` (options can be combined)
  Long listing, most recent files at the end

VLAAMS
SUPERCOMPUTER
CENTRUM

ls command

- `$ ls *txt`
  The shell first replaces *txt by all the file and directory names ending by txt (including .txt), except those starting with ., and then executes the ls command line.

- `$ ls –d .*`
  Lists all the files and directories starting with .
  -d tells ls not to display the contents of directories.
  `$ ls –d */`

- `$ ls ?.log`
  Lists all the files which names start by 1 character and end by .log

- http://www.thegeekstuff.com/2009/07/linux-ls-command-examples/

ls command

```
vsc30051@login1:~/matlab/test> ls -al
total 1156
drwxr-xr-x 9 vsc30051 vsc30051   8192 2011-03-29 16:10 .
drwxr-xr-x 9 vsc30051 vsc30051   4096 2011-06-20 12:08 ..
drwxr-xr-x 3 vsc30051 vsc30051   4096 2010-01-08 10:54 courseAndreas
-rwxr--r-- 1 vsc30051 vsc30051 149326 2011-03-10 15:14 fibonacci
-rw-r--r-- 1 vsc30051 vsc30051    681 2011-03-10 15:09 fibonacci.m
-rwxr--r-- 1 vsc30051 vsc30051 149068 2011-03-10 12:16 ftest
-rw-r--r-- 1 vsc30051 vsc30051    443 2011-03-10 12:17 ftest_compiled.sh
-rw-r--r-- 1 vsc30051 vsc30051    442 2011-03-10 11:28 ftest_compiled.sh~
-rw-r--r-- 1 vsc30051 vsc30051     70 2011-03-10 12:15 ftest.m
```

- In Linux, file is defined as simply the thing that deals with a sequence of bytes
- Hence everything are files: an ordinary file is a file; a directory is also file; a network card, a hard disk, any device are also files since they deal with a sequence of bytes

VLAAMS
SUPERCOMPUTER
CENTRUM

Moving around

- **Symbolic (soft) link**
  - Not a real file, just a link to another file
  - Allows giving another name to a file without actually duplicating it – hence saves memory space

- **Special file (device)**
  - Each hardware device, e.g. keyboard, hard disk, CD-ROM, etc. is associated with at least one file
  - Usually store in /dev directory
  - Applications can read and write any devices by reading and writing their associate file – hence the access method is known as device independent
  - Divide into two types: character special files, e.g. keyboard, and block special files, e.g. disk

ls command

Command that sets a symbolic link to a file called CUI to anotherCUI

Names in blue are directories, indicated by a letter d at the beginning of the line

A symbolic link begins with a letter l

```
dlun@enpklun.polyu.edu.hk: /home/dlun/Desktop

File   Edit   Settings   Help

[dlun@enpklun Desktop]$ ln -s ../CUI anotherCUI
[dlun@enpklun Desktop]$ ls -al
total 44
drwxr-xr-x     5 dlun      dlun          4096 Jan   4 18:36 .
drwx------    16 dlun      dlun          4096 Jan   4 18:26 ..
drwxr-xr-x     2 dlun      dlun          4096 May 17  2001 Autostart
-rw-r--r--     1 dlun      dlun           230 May 17  2001 Printer.kdelnk
-rw-r--r--     1 dlun      dlun           159 May 17  2001 Red Hat Errata.kdelnk
-rw-r--r--     1 dlun      dlun           153 May 17  2001 Red Hat Support.kdelnk
drwxr-xr-x     2 dlun      dlun          4096 May 17  2001 Templates
drwxr-xr-x     2 dlun      dlun          4096 May 17  2001 Trash
lrwxrwxrwx     1 dlun      dlun             6 Jan   4 18:36 anotherCUI -> ../CUI
-rw-r--r--     1 dlun      dlun           388 May 17  2001 cdrom.kdelnk
-rw-r--r--     1 dlun      dlun           395 May 17  2001 floppy.kdelnk
-rw-r--r--     1 dlun      dlun           144 May 17  2001 www.redhat.com.kdelnk
[dlun@enpklun Desktop]$ 
```

VLAAMS
SUPERCOMPUTER
CENTRUM

Moving around

- Display the current/working directory
  - `$ pwd`
  - **P**rint **W**orking **D**irectory
  - displays your current location within the file system.
- Change (navigate) directories.
  - `$ cd dir_name`
  - **C**hange **D**irectories
  - changes the position to the specific directory
- You can specify directory names in two ways:
  - Absolute pathname (starts from the root of the tree)
    `$ cd /u/home/hpc/test/bin`
  - Relative pathname (relative to your current directory)
    `$ cd`
    `$ cd .`
    `$ cd ..`
    `$ cd test/bin`

VLAAMS
SUPERCOMPUTER
CENTRUM

Special directories

- .
  - The current directory.
  - Useful for commands taking a directory argument.
  - Useful to run commands in the current directory
  - ./readme.txt and readme.txt are equivalent.
- ..
  - The parent (enclosing) directory. Always belongs to the .
    Directory
  - Typical usage:
    cd ..
  - ~
    - Shells just substitute it by the home directory of the current
      user.
  - -
    - cd – jump back to the previous directory

Linux File System - directories
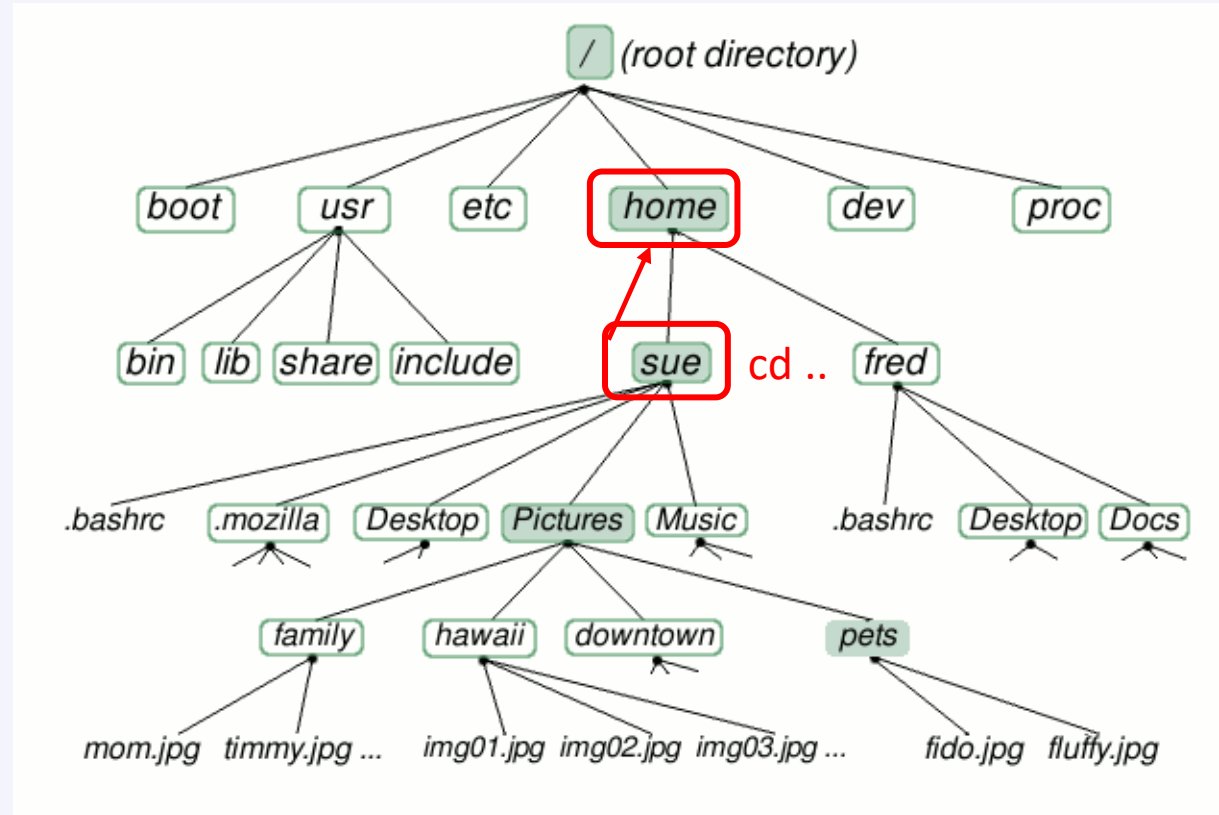
Source: http://www.linuxplanet.com/linuxplanet/tutorials/6666/1/screenshot3894/

Linux File System - directories

Source: http://www.linuxplanet.com/linuxplanet/tutorials/6666/1/screenshot3894/

Linux File System - directories

Source: http://www.linuxplanet.com/linuxplanet/tutorials/6666/1/screenshot3894/

Linux File System - directories

Source: http://www.linuxplanet.com/linuxplanet/tutorials/6666/1/screenshot3894/

Linux File System - directories

Source: http://www.linuxplanet.com/linuxplanet/tutorials/6666/1/screenshot3894/

Linux File System - directories

Source: http://www.linuxplanet.com/linuxplanet/tutorials/6666/1/screenshot3894/
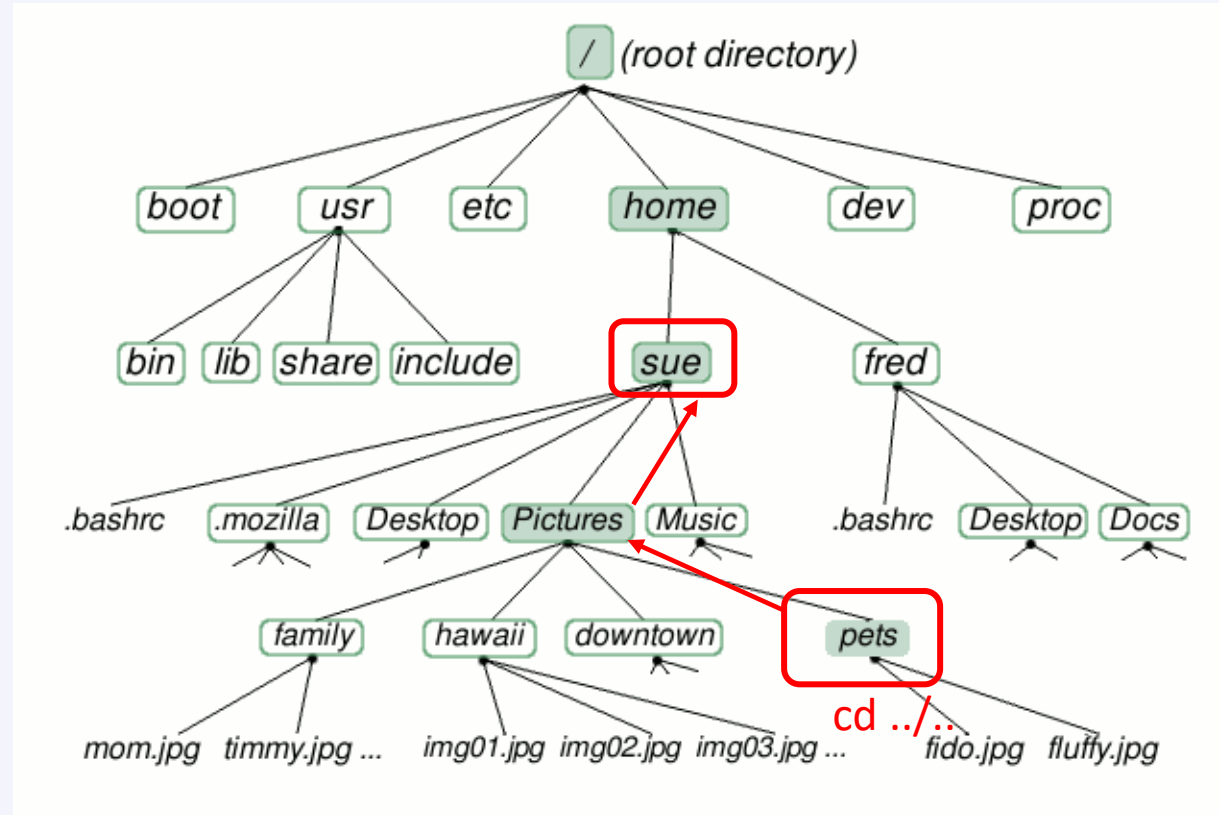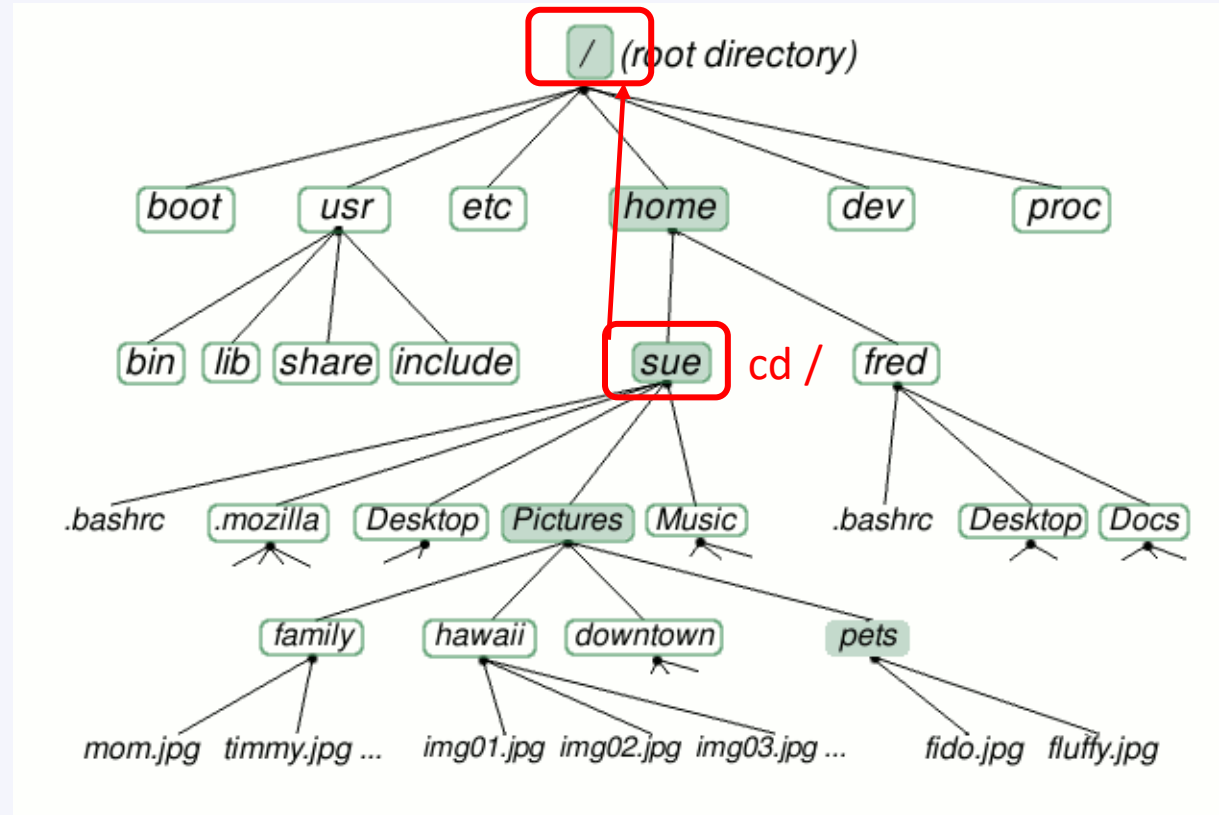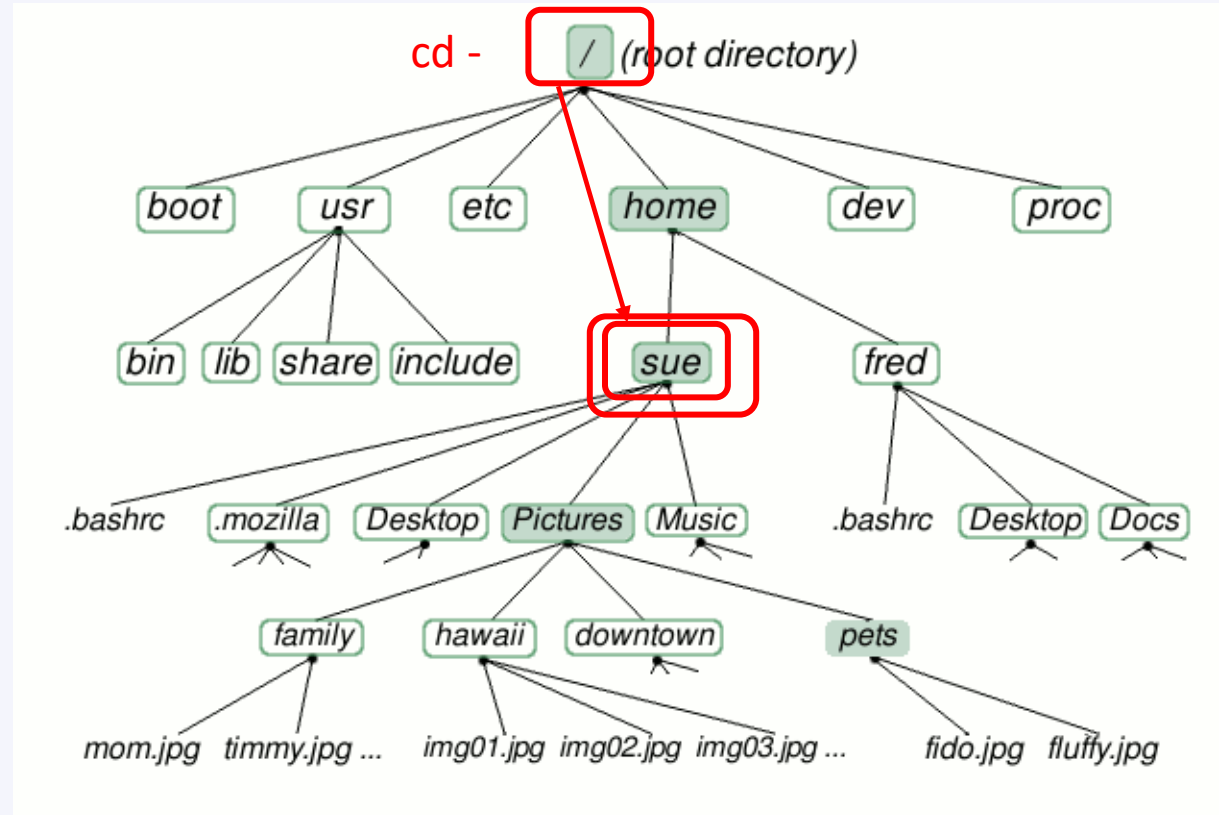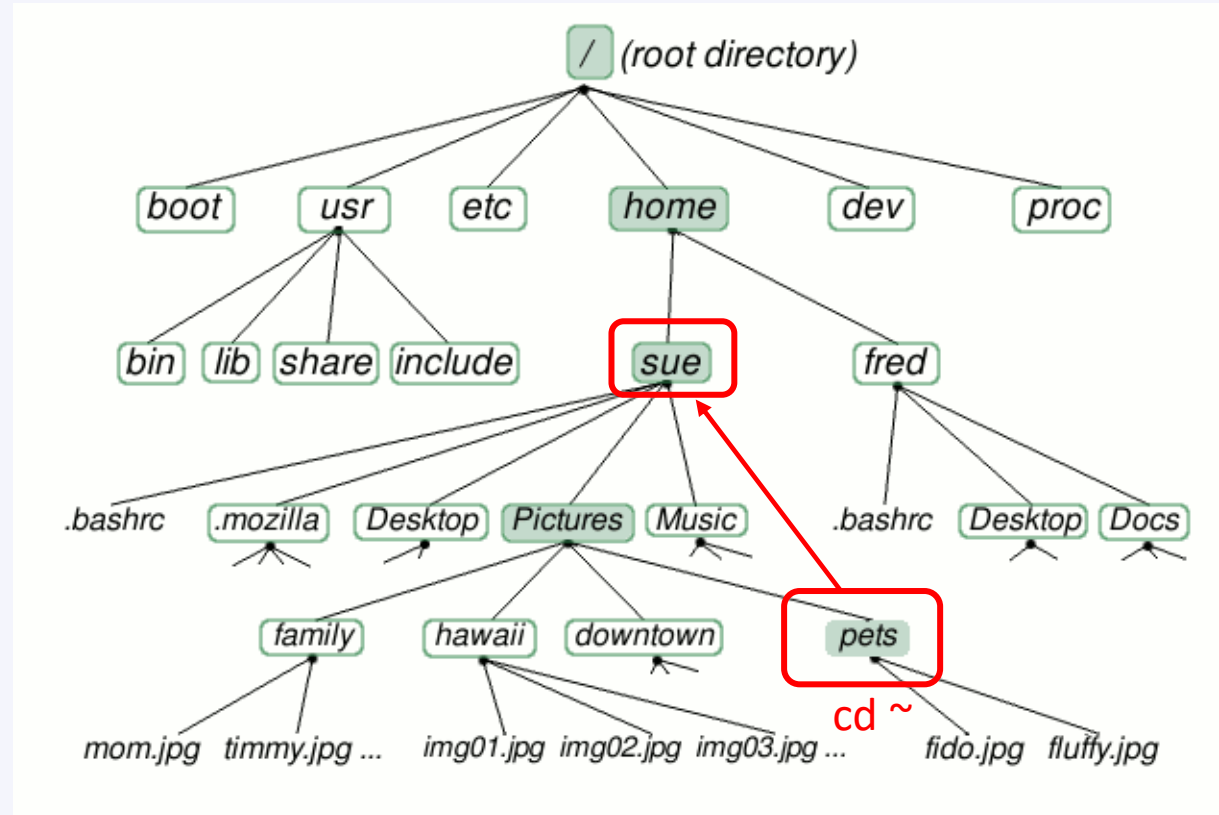
Linux File System-home directory

Relative path

cd ../Desktop

Source: http://www.linuxplanet.com/linuxplanet/tutorials/6666/1/screenshot3894/
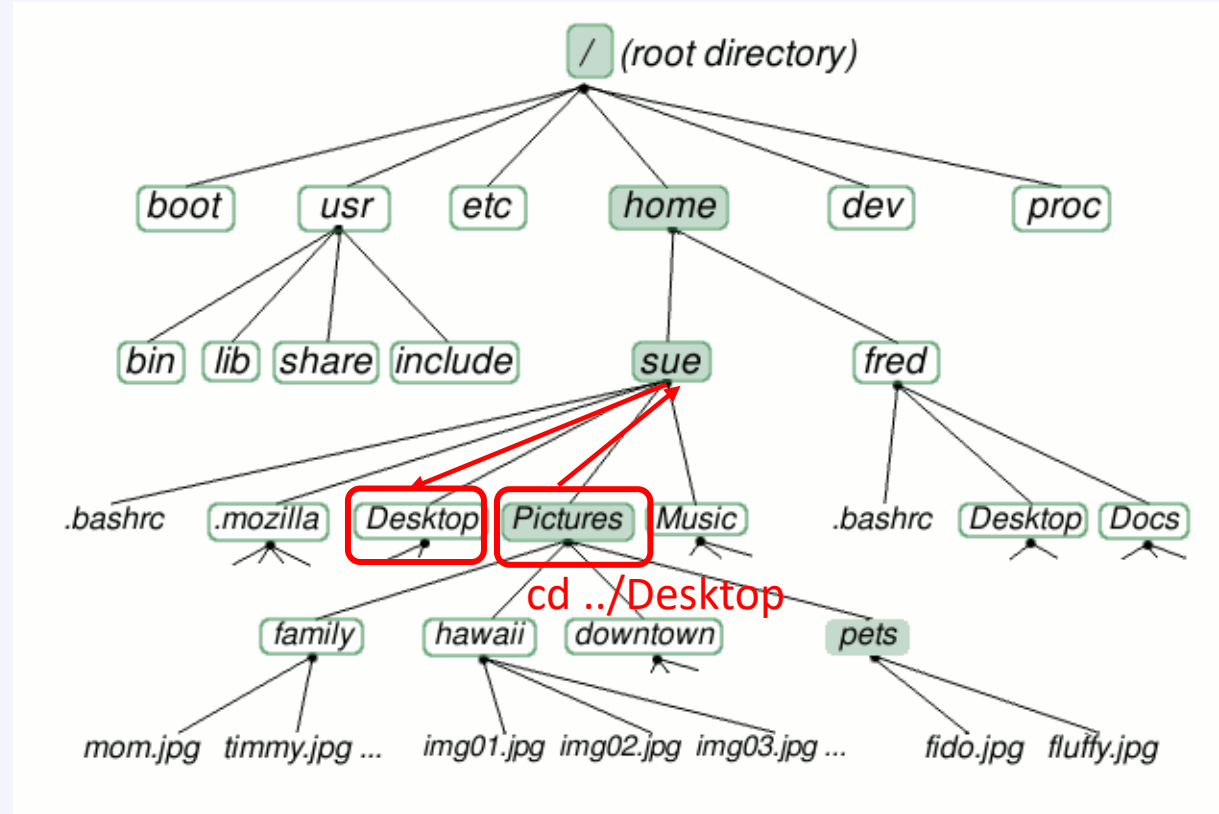
Linux File System - directories

Relative path

cd ~/Desktop

Source: http://www.linuxplanet.com/linuxplanet/tutorials/6666/1/screenshot3894/

Linux File System - directories

Source: http://www.linuxplanet.com/linuxplanet/tutorials/6666/1/screenshot3894/
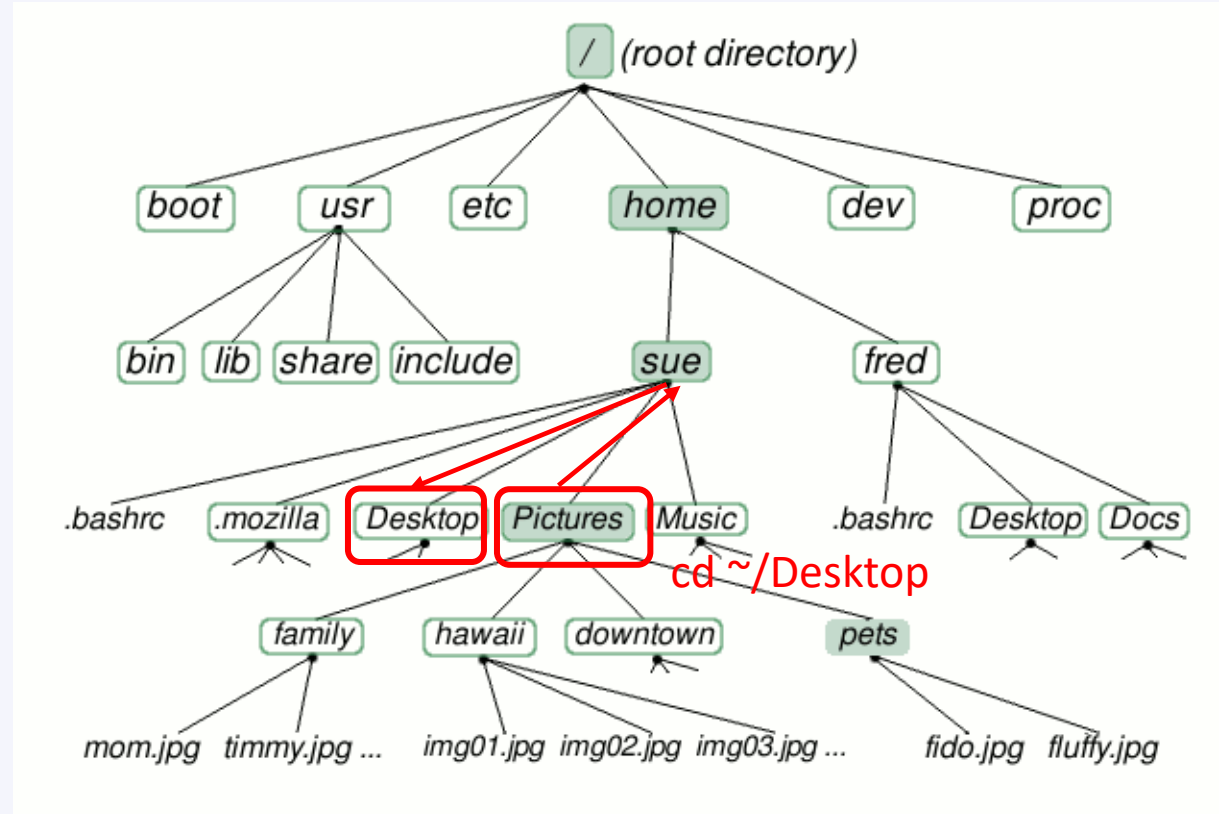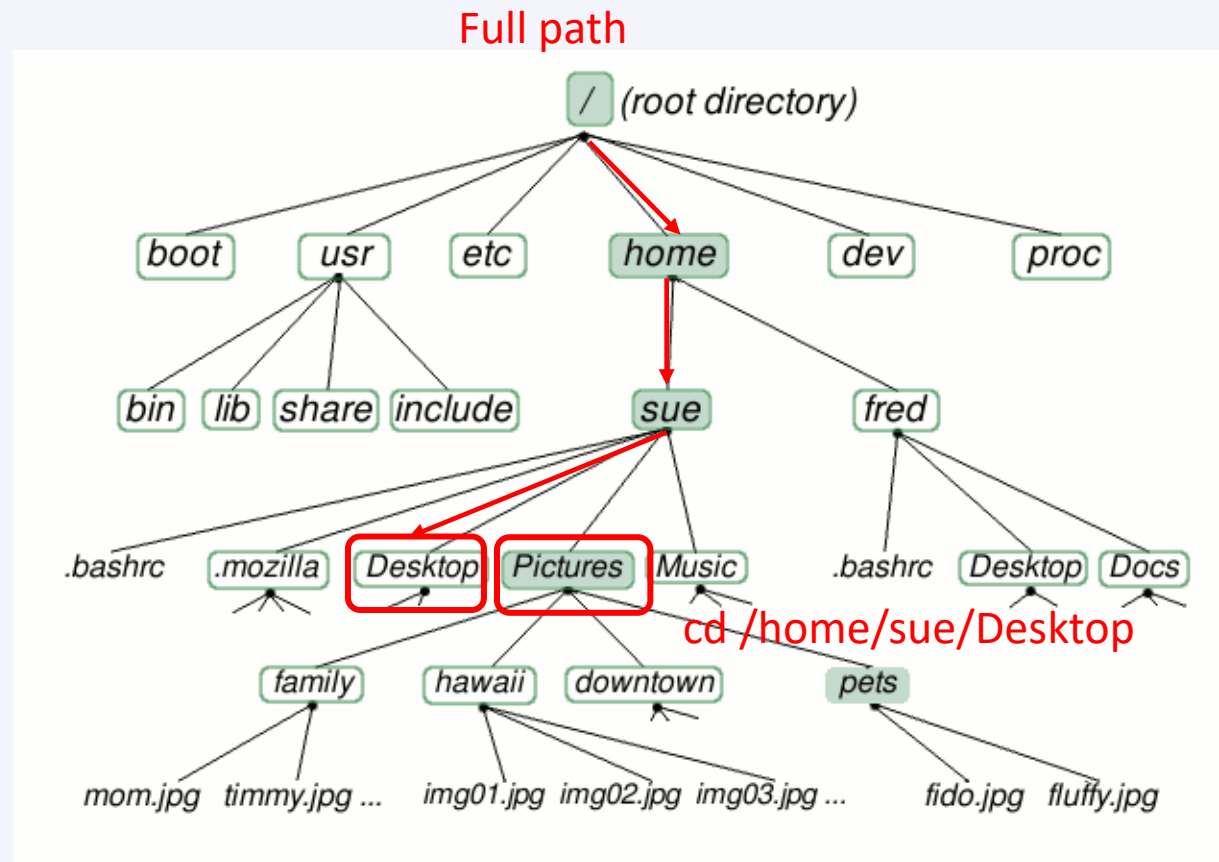
File paths

- A path is a sequence of nested directories with a file or directory at the end, separated by the / character
- **Relative path**: `documents/fun/file1`
  Relative to the current directory
- To work with relative paths they need to have connection on the linux tree
- **Absolute path**: `/home/user/leuven/file2`
- `/` : root directory.
  Start of absolute paths for all files on the system
    - Not a superuser's home directory (-> /root)

More on files

- What does a file contain?
    - Determine a file's type: `file`
    - will print a brief description of the file's contents
    - `$ file  filename`
- Search for files and directories
    - The **find** command performs a raw search on a file system to locate the specified items.
    - `$ find  location -name some-name`
      `($ find /home/student -name *desktop)`
    - Be careful: results can sometimes be slow
- Search the locate database for files and directories
    - The `locate` command displays the location of files that match the specified name.
    - Faster than find because it searches a database of indexed filenames.
    - Disadvantage: lacks the ability to search for advanced characteristics such as file owner, size, and modification time.

File names

File name features:

- Case sensitive
- No obvious length limit (255 characters)
- Can contain any character (including whitespace, except /).
- File name extensions not needed and not interpreted. Just used for user convenience. (File types stored in the file)
- a hidden file is any file that begins with a "." (not seen with the bare *ls*
- File name examples:
  ```
  README        .bashrc     index.htm index.html
     index.html.old
  ```

Auto-Completion

- Have the shell automatically complete commands or file paths.
- Activated using the **<TAB>** key on most systems
- examples
  - `$ whe<TAB>`
  - `$ whereis`
  - `$ ls -l /etc/en<TAB>`
  - `$ ls -l /etc/environment`
- When more than one match is found, the shell will display all matching results (use **<TAB>** twice)
  - `$ ls -l /etc/host<TAB>`

VLAAMS
SUPERCOMPUTER
CENTRUM

Command history: Arrow Up

- Previously executed commands can be recalled by using the **Up Arrow** key on the keyboard.

- Most Linux distributions remember the last five hundred commands by default.

- Display commands that have recently been executed
  - The `history` command displays a user's command line history.
  - You can execute a previous command using `![NUM]` where NUM is the line number in history you want to recall.

Globbing: use wildcard

Wildcard	Function
*	Matches 0 or more characters
?	Matches 1 character
[abc]	Matches one of the characters listed
[a-c]	Matches one character in the range
[!abc]	Matches any character not listed
[!a-c]	Matches any character not listed in the range
{tacos,nachos}	Matches one word in the list

```
$ ls -l /etc/host*
$ ls -l /etc/hosts.{allow,deny}
$ ls -l /etc/hosts.[!a]*
$ ls -l /etc/host?
```

Input and Output
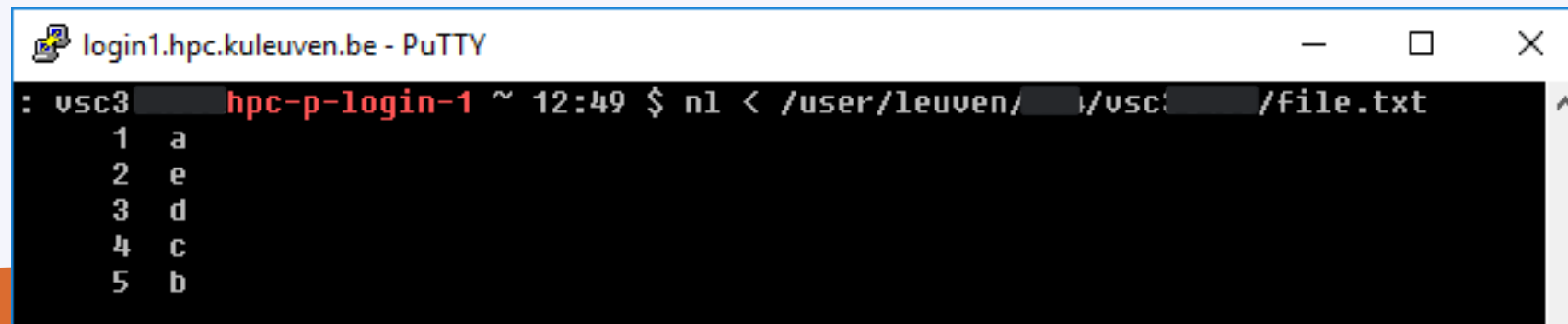
- Programs and commands can contain an input and output. These are called 'streams'. UNIX programming is oftentimes stream based.

- STDIN – 'standard input,' or input from the keyboard

- SDTOUT – 'standard output,' or output to the screen

- STDERR – 'standard error,' error output which is sent to the screen.

VLAAMS
SUPERCOMPUTER
CENTRUM

File Redirection

- Often we want to save output (stdout) from a program to a file. This can be done with the 'redirection' operator.

    - **myprogram** *> myfile* – using the '>' operator we redirect the output from **myprogram** to file *myfile*

- Similarly, we can append the output to a file instead of rewriting it with a double '>>'

    - **myprogram** *>> myfile* – using the '>' operator we append the output from **myprogram** to file *myfile*

Input Redirection

- Input can also be given to a command from a file instead of typing it to the screen, which would be impractical.

  - **mycommand** < *programinput* – using the '< ' operator we redirect the input from the file *programinput* to **mycommand**
  - *programinput* is printed to stdout, which is redirected to a command **mycommand.**
  - Not all commands read standard input (ls, date, who, pwd, cd, ps, …)

Redirecting stderr

- Performing a normal redirection will not redirect sdterr. In Bash, this can be accomplished with '2>'
  - **command** *2> file1*

- Or, one can merge stderr to stdout (most popular) with '2>&1'
  - **command** *> file 2>&1*

VLAAMS
SUPERCOMPUTER
CENTRUM

Pipes

- Using a pipe operator '|' commands can be linked together. The pipe will link the standard output from one command to the standard input of another.

- Very helpful for searching files

- e.g. when we want to list the files, but only the ones that contain test in their name:
  ```
  ls -la|grep test
  ```

Pipes

- Lots of Little Tools

```
echo "Hello" | \
  wc -c
```
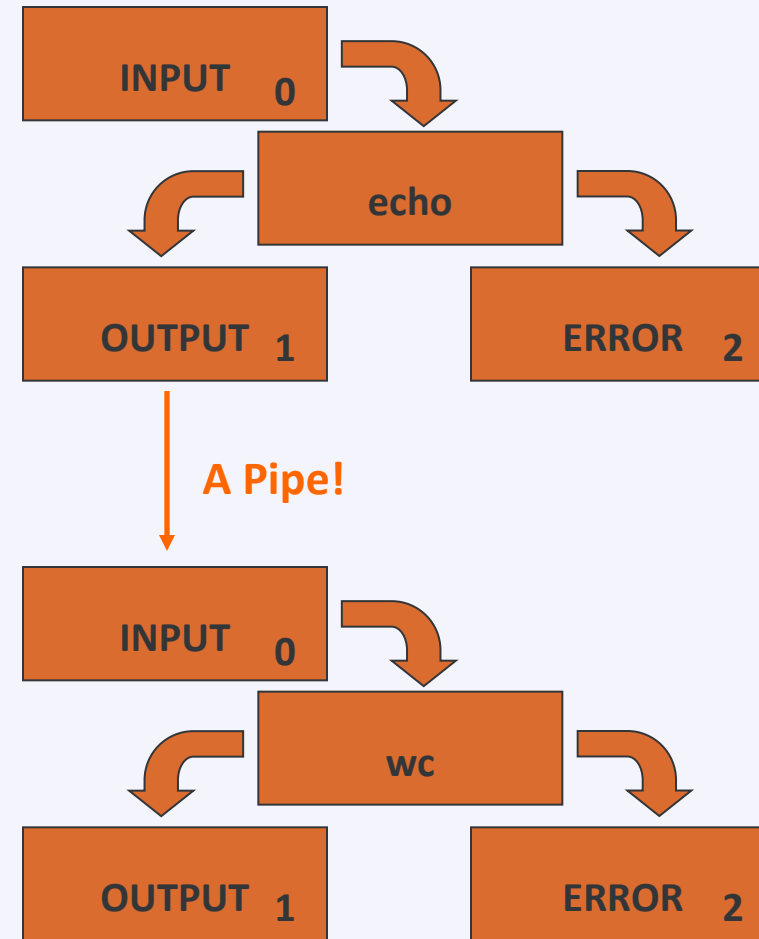
Difference Between Single, Double, and Backwards Quote

- Single quotes (') do not interpret any variables

- Double quotes (") interpret variables

- Backwards quotes (`) interpret variables and treat them as a program to run and return the results of that program

VLAAMS
SUPERCOMPUTER
CENTRUM

Quote characters

There are three different quote characters with different behaviour. These are:

" : double quote, weak quote. If a string is enclosed in " " the references to variables (i.e $variable) are replaced by their values. Also, back-quote and escape \ characters are treated specially.

' : single quote, strong quote. Everything inside single quotes are taken literally; nothing is treated as special.

` : back quote. A string enclosed as such is treated as a command and the shell attempts to execute it. If the execution is successful, the primary output from the command replaces the string.

Example: `echo "Today is:" `date``

File Manipulation

- For all file manipulation commands relative of absolute paths can be used (or just files in the current directory when no extra path specified)

- Most of the commands are intuitive – shortcuts of English names

VLAAMS
SUPERCOMPUTER
CENTRUM

Directories

- Create directories
  - The `mkdir` command is used to create directories
  - `$ mkdir dir1 dir2 dir3`
- Remove directories
  - The `rmdir` command removes directories
  - `$ rmdir dir1`
  - `rmdir` will only remove empty directories.
    To remove a non-empty directory, use
    `$ rm -r [DIRECTORY]` instead.

VLAAMS
SUPERCOMPUTER
CENTRUM

Copy a file

- The `cp` command copies files and directories
- The default behavior will overwrite any existing file(s). The **-i** option overrides this behavior and prompts the user before overwriting the destination file.
- **syntax: `cp [OPTIONS] [SOURCE] [DESTINATION]`**
  - `$ cp <source_file> <target_file>`
    Copies the source file to the target.
  - `$ cp file1 file2 file3 ... dir`
    Copies the files to the target directory (last argument).
  - `$ cp -i` (interactive)
    Asks for user confirmation if the target file already exists
  - `$ cp -r <source_dir> <target_dir>` (recursive)
    Copies the whole directory.
  - `$ cp -v` (verbose)
    Displays what has been copied

Move or rename files

- Move or rename files and directories: `mv`
- The default behavior will overwrite any existing file(s).
- **syntax: `mv [OPTIONS] [SOURCE] [DESTINATION]`**
  - `$ mv old_name new_name`
    Renames the given file or directory.
  - `$ mv -i` (interactive)
    If the new file already exits, asks for user confirm
- The `mv` command can also be used to move or rename directories
  - `$ mv NewFiles/ OldFiles/`
  - -r option is not necessary

Remove files

- The rm command removes files.

- `$ rm file1 file2 file3 ...`
  Removes the given files.

- `$ rm -i` (interactive)
  Always ask for user confirmation.

- `$ rm -r dir1 dir2 dir3` (recursive)
  Removes the given directories with all their contents.

- Tip:
  Whenever you use wildcards with `rm` (besides carefully checking your typing!), test the wildcard first with `ls`. This will let you see the files that will be deleted. Then press the up arrow key to recall the command and replace the `ls` with `rm`.

VLAAMS
SUPERCOMPUTER
CENTRUM

Create links

- Create links (shortcuts) to files or directories: `ln`
- Symbolic links are created when using the `-s` option with the `ln` command.
- Allows to jump to other files or locations on the file system
- Editing a symbolic link file is the same as editing the source file, but deleting the symbolic link does not delete the source file.
  - `$ ln -s file_v5.doc file_final.doc`
  - creates a symbolic link called file_final.doc that points to file_v5.doc
  - `$ ln -s /home/demo/dir1/dir2/dir3 /home/demo/jump2dir`
  - creates a symbolic link called jump2dir that points to a deep directory - allows for quicker access)

Text editors

- Text-only text editors
  - Often needed for sysadmins and great for power users
  - vi, vim
  - nano
- Graphical text editors
  - Fine for most needs
  - Gedit,
  - Kate, Nedit
  - Emacs, Xemacs
- http://www.thegeekstuff.com/2009/07/top-5-best-linux-text-editors/

VLAAMS
SUPERCOMPUTER
CENTRUM

Text editors

How to start:

- Create a file
  - `$ touch filename` (creates an empty file)
  - start editing (non)-existing file:

    `$ vi filename`

    `$ nano filename`

    `$ gedit filename`

vi

- Text-mode text editor available in all Linux systems.
- Created before computers with mice appeared.
- Difficult to learn for beginners used to graphical text editors.
- Very productive for power users.
- Check the web for tutorials:
  - https://upload.wikimedia.org/wikipedia/commons/d/d2/Learning_the_vi_Editor.pdf
  - ftp://ftp.vim.org/pub/vim/doc/book/vimbook-OPL.pdf

vi

- 2 basic modes of operation:
  - *command mode* and *editing mode*.
  - Within Command Mode, signals from the terminal are interpreted as editing commands.
  - Editing mode: letters typed at the keyboard are inserted into the editing buffer.
- Pressing **Esc** on the keyboard activates command mode.

Key(s)	Function	Key(s)	Function
:w	Save	A	Append text after
:x	Save and exit	r	Replace text before cursor
:q	Quit	R	Replace text after cursor
I	Insert text after	i	Insert text before
P	Paste copied text	yy	Copy current line
a	Append text before	/[TEXT]	Search for the specified text

Vi

- 2 modes
  - Input mode
    - ESC to back to cmd mode
  - Command mode
    - Cursor movement
      - h (left), j (down), k (up), l (right)
      - ^f (page down)
      - ^b (page up)
      - ^ (first char.)
      - $ (last char.)
      - G (bottom page)
      - :1 (goto first line)
    - Switch to input mode
      - a (append)
      - i (insert)
      - o (insert line after
      - O (insert line before)

- Delete
  - dd (delete a line)
  - d10d (delete 10 lines)
  - d$ (delete till end of line)
  - dG (delete till end of file)
  - x (current char.)
- Paste
  - p (paste after)
  - P (paste before)
- Undo
  - u
- Search
  - /
- Save/Quit
  - :w (write)
  - :q (quit)
  - :wq (write and quit)
  - :q! (give up changes)

VLAAMS
SUPERCOMPUTER
CENTRUM

Nano
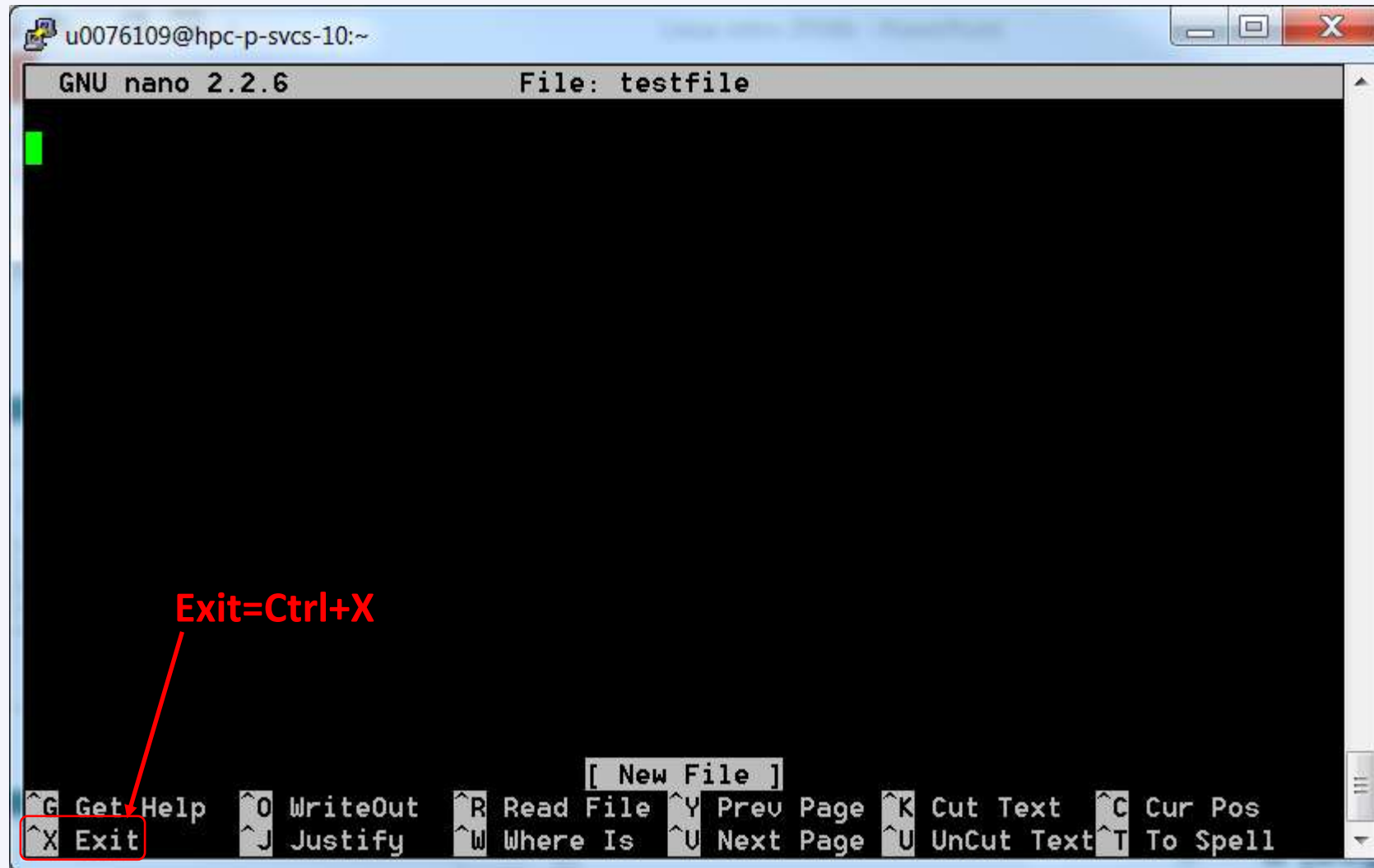
Displaying file contents

Several ways of displaying the contents of files.

- `$ cat file1`
  displays the contents of the given file.

- `$ cat file1 file2 file3 ...` (concatenate)
  Concatenates and outputs the contents of the given files.

- `$ more file1`
  Display the output of a command or text file one page at a time.

- Can also jump to the first occurrence of a keyword
  (/ command).

Displaying file contents

- `$ less file1`
  Does more than more.
  Doesn't read the whole file before starting.
  Supports backward movement in the file (? command).
    Press `q` to exit

- `$ display file1`
    Displays graphical file (simple image)

VLAAMS
SUPERCOMPUTER
CENTRUM

The head and tail commands

- `$ head [-<n>] <file>`
  Displays the first <n> lines (or 10 by default) of the given file.
  Doesn't have to open the whole file to do this!

- `$ tail [-<n>] <file>`
  Displays the last <n> lines (or 10 by default) of the given file.
  No need to load the whole file in RAM! Very useful for huge files.

- `$ tail -f <file>` (follow)
  Displays the last 10 lines of the given file and continues to display new lines when they are appended to the file.
  Very useful to follow the changes in a log file, for example.

The grep command

- `$ grep <pattern> <files>`
  Scans the given files and displays the lines which match the given pattern.

- `$ grep error *.log`
  Displays all the lines containing error in the *.log files

- `$ grep –i error *.log`
  Same, but case insensitive

- `$ grep –ri error .`
  Same, but recursively in all the files in the current directory and its subdirectories

- `$ grep –v info *.log`
  Outputs all the lines in the files except those containing info.

- http://www.thegeekstuff.com/2009/03/15-practical-unix-grep-command-examples/

VLAAMS
SUPERCOMPUTER
CENTRUM

wget

- Instead of downloading files from your browser, just copy and paste their URL and download them with wget

- main features
  - http and ftp support
  - Can resume interrupted downloads
  - Can download entire sites or at least check for bad links
  - Very useful in scripts or when no graphics are available
    (system administration, embedded systems)

- `$ wget -c http://microsoft.com/customers/dogs/winxp4dogs.zip`
Continues an interrupted download.

- `$ wget -r -np http://www.xml.com/ldd/chapter/book/`
Recursively downloads an on-line book for off-line access.
-np: "no-parent". Only follows links in the current directory.

Measuring disk usage

- `$ du -h <file>`
  -h: returns size on disk of the given file, in human readable format: K (kilobytes), M (megabytes) or G (gigabytes),
  Without -h, du returns the raw number of disk blocks used by the file (hard to read).
  Note that the -h option only exists in GNU du.

- `$ du -sh <dir>`
  -s: returns the sum of disk usage of all the files in the given directory.

VLAAMS
SUPERCOMPUTER
CENTRUM

File Archiving: tar

- Saves and restores multiple files to/from a single file. Directories followed recursively.
- Format:
    - `$ tar [options] [options_values] [files]`

    - c – create a new archive
    - v – verbosely list files which are processed.
    - f – following is the archive file name
    - z – filter the archive through gzip (compress)
    - x – extract files from archive
    - j - filter the archive through bzip (compress)

File Archiving: tar

- Examples:
  - `$ tar -cvf [FILE] [ITEM]`   Backup the specified item(s)
    `$ tar -cvf /tmp/backup.tar ~/data ~/test`
  - `$ tar -czvf [FILE] [ITEM]`   Compress the archive to save space
  - `$ tar -xvf [FILE] [ITEM]`   Restore the specified item(s) **`$tar -xvf backup.tar`**
  - `$ tar -tf [FILE]`   List all files in the specified archive
    e.g. `$ tar -tf backup.tar`
- http://www.thegeekstuff.com/2010/04/unix-tar-command-examples/

VLAAMS
SUPERCOMPUTER
CENTRUM

File Compression: gzip

- Compressing files: `gzip filename` or `bzip2 filename`
  - `$ gzip backup.tar`
  - `$ bzip2 backup.tar`

  The resulted file is backup.tar.gz/ backup.tar.bz2

- Uncompressing files: `gzip -d filename.gz` or `bzip2 -d filename.bz2`
  - `$ gzip -d backup.tar.gz`
  - `$ bzip2 -d backup.tar.bz2`

  The uncompressed file is backup.tar

VLAAMS
SUPERCOMPUTER
CENTRUM

File access rights

Linux File Access Privilege

- Linux is a multiuser system, the files of all users are stored in a single file structure

- Mechanism is required to restrict one user to access the files of another user, if he is not supposed to

- User can impose access permission to each file to restrict its access

- The term "access permission" refers to

  o read permission

  o write permission

  o execute permission

File access rights

3 types of **access rights**

- Read access (r)
  - reading, opening, viewing, and copying the file is allowed
- Write access (w)
  - writing, changing, deleting, and saving the file is allowed
- Execute rights (x)
  - executing and invoking the file is allowed. This is required for directories to allow searching and access.

Use `ls -l` *to check file access rights*

VLAAMS
SUPERCOMPUTER
CENTRUM

File access rights

3 types of **access levels**

- User (u): for the owner of the file
- Group (g): each file also has a "group" attribute, corresponding to a given list of users
- Others (o): for all other users

File access rights

Hard link no

Owner

Owner's group

File last modified date

d rwx r-x r-x   2   dlun dlun 4096 May 17 2001 Autostart

It is a directory

file size

file name

The directory can be read, written and executed by the user dlun

The directory can be read and executed but not written by other users in the same group of dlun

The directory can be read and executed but not written by other users in different group of dlun

The group of a user is assigned by the administrator when a user is added to the system

VLAAMS
SUPERCOMPUTER
CENTRUM

File access rights

- Access permission can also be assigned to a directory
- Directory is also a file that contains the attributes of the files inside it
- If read permission is not given to a directory
  - cannot show the structure of this directory
  - e.g. cannot use ls
- If write permission is not given to a directory
  - cannot modify anything of the directory structure
  - e.g. cannot copy a file into this directory since it will modify the directory structure by adding one more file
- If execute permission is not given to a directory
  - nearly nothing can be done with this directory, even cd

Access rights examples

- -rw-r--r--
  Readable and writable for file owner, only readable for others

- -rw-r-----
  Readable and writable for file owner, only readable for users belonging to the file group.

- drwx------
  Directory only accessible by its owner

- -------r-x
  File executable by others but neither by your friends nor by yourself. Nice protections for a trap...

VLAAMS
SUPERCOMPUTER
CENTRUM

Access rights examples

temp does not have execution right

even cd is not workable

execution right is added

now we can change the directory to temp

chmod: changing permissions

- Permissions allow you to share files or directories or to lock them down to be private.

- `$ chmod` (change mode)

- `$ chmod <permissions> <files>`
  2 formats for permissions:
    - octal format (3 digit octal form)
    - symbolic format

chmod: changing permissions

- octal format (abc):
  a,b,c = r*4+w*2+x*1 (r, w, x: booleans)
  - 0 none   ---
  - 1 execute-only   --x
  - 2 write   -w-
  - 3 execute and write   -wx
  - 4 read-only   r—
  - 5 read and execute   r-x
  - 6 read and write   rw-
  - 7 read, write, and execute   rwx

  - `$ chmod 644 <file>`
    (rw for u, r for g and o)

660 : 110 110 000
$\Rightarrow$ rw- rw- ---
545 : 101 100 101
$\Rightarrow$ r-x r-- r-x

chmod: changing permissions

- symbolic format:
  $ chmod go+r: add read permissions to group and others.
  $ chmod u-w: remove write permissions from user.
  $ chmod a-x: (a: all) remove execute permission from all.
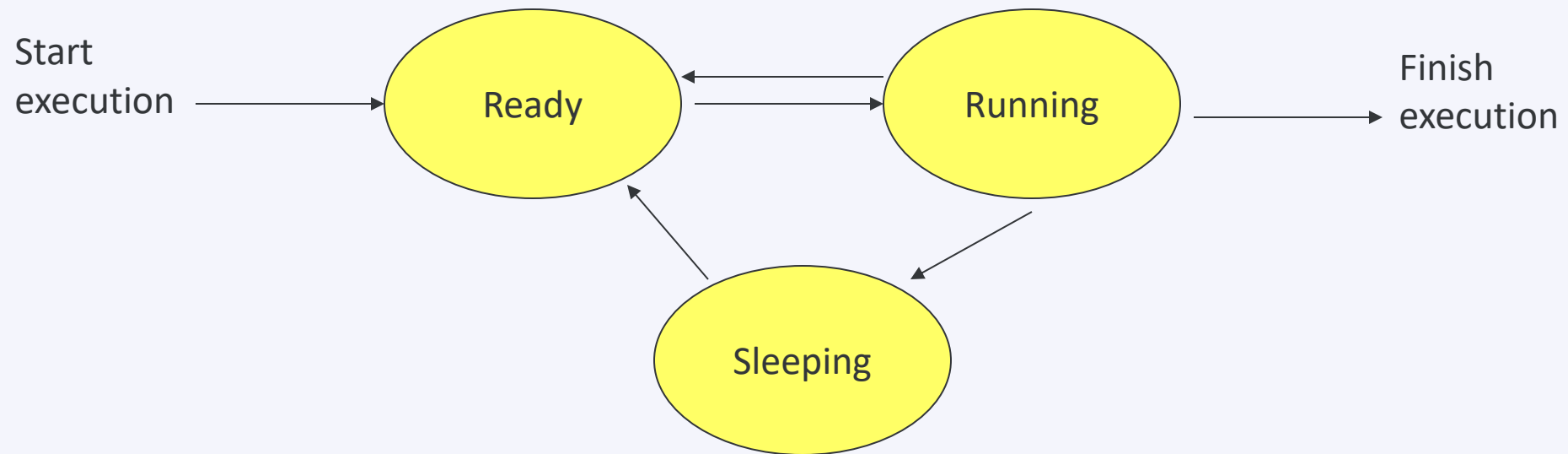
Access right constraints

- x is sufficient to execute binaries
  Both x and r and required for shell scripts.

- Both r and x permissions needed in practice for directories:
  r to list the contents, x to access the contents.

- You can't rename, remove, copy files in a directory if you don't have w access to this directory.

- If you have w access to a directory, you can remove a file even if you don't have write access to this file (remember that a directory is just a file describing a list of files). This even lets you modify (remove + recreate) a file even without w access to it.

VLAAMS
SUPERCOMPUTER
CENTRUM

Process

- "Everything in Unix is a file. Everything in Unix that is not a file is a process"
- Processes
  - Instances of a running programs
  - Several instances of the same program can run at the same time
  - Processes are assigned a unique identifier which is used to monitor and control the process (PID)

VLAAMS
SUPERCOMPUTER
CENTRUM

Process

- A program that is claimed to be executing is called a process
- For a multitasking system, a process has at least the following three states:

Start execution →

Ready ⇄ Running → Finish execution

Running → Sleeping → Ready

VLAAMS
SUPERCOMPUTER
CENTRUM

Process

- Ready state
  - All processes that are ready to execute but without the CPU are at the ready state
  - If there is only 1 CPU in the system, all processes except one are at the ready state
- Running state
  - The process that actually possesses the CPU is at the running state
  - If there is only 1 CPU in the system, at most there is only one process is at the running state
- Sleeping state
  - The process that is waiting for other resources, e.g. I/O, is at the sleeping state

VLAAMS
SUPERCOMPUTER
CENTRUM

ps

- Display running processes
  (cfr. Windows Task Manager ctrl-shift-esc)
- `$ ps`   Display the current user's processes
- `$ ps -e`   Display all processes running on the system
- `$ ps -ef`   Display detailed information about running processes
- `$ ps -u [USER]`   Display processes owned by the specified user
- `$ ps a`   Display extra info (running state)

VLAAMS
SUPERCOMPUTER
CENTRUM

Process

```
  PID   TTY      STAT    TIME COMMAND
14748  pts/1  S       0:00 -bash
14795  pts/0  S       0:00 -bash
14974  pts/0  S       0:00 vi test1.txt
14876  pts/1  R       0:00 ps …
```

Process ID

Terminal name

State:
S – Sleeping
(waiting for input)
R – Running

How much time the process is continuously executing

- For the example above, both bash processes, which are the shell of both terminals, are waiting for the input of user. They must be in the sleeping state

- The vi process, which is an editor, is also waiting for the input of user. Hence it is also in sleeping state

- When ps reporting the processes in the system, it is the only process that is running. Hence it is in running state

VLAAMS
SUPERCOMPUTER
CENTRUM

kill

- Sends an abort signal to the given processes. Lets processes save data and exit by themselves. Should be used first.

- `$ kill <pid>`
  Example:
  `$ kill 3039 3134 3190 3416`

- `$ kill -9 <pid>`
  Sends an immediate termination signal. The system itself terminates the processes. Useful when a process is really stuck.

&

- **&** is a command line operator that instructs the shell to start the specified program in the background.

- This allows you to have more than one program running at the same time without having to start multiple terminal sessions.

- Starting a process in background: add & at the end of your line:
  `$ gedit &`
  check with `ps`

VLAAMS
SUPERCOMPUTER
CENTRUM

top

- Displays most important processes, sorted by cpu percentage (use > or < to change the order)

  http://www.thegeekstuff.com/2010/01/15-practical-unix-linux-top-command-examples/

List of basic commands

- Managing files and directories:
  `cp, mv, rm, mkdir, rmdir, touch`
- Displaying content of files
  `cat, more, less, head, tail`
- Moving around
  `pwd, cd, ls`
- Asking for help
  `help, man, info, whatis`

  + understanding of Linux directory tree (relative and absolute paths)
  + understanding of file permissions

VLAAMS
SUPERCOMPUTER
CENTRUM

SSH overview

Private keys are always secret

- Anyone who can access your private key can log in as you!
- Set a passphrase on your private key
- Private key is encrypted with this passphrase
- Always a pair of keys is needed
- Both keys need to be generated together

VLAAMS
SUPERCOMPUTER
CENTRUM

Generate the key

Linux/Mac OS users:

- Use *"ssh-keygen "* command to generate key pair
- **Be sure to give your key a passphrase!**
- Requested ssh key format: RSA 4096 bit

```
user@desktop:~> ssh-keygen -t rsa -b 4096
Generating public/private rsa key pair.
Enter file in which to save the key (/home/user/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/user/.ssh/id_rsa.
Your public key has been saved in /home/user/.ssh/id_rsa.pub.
The key fingerprint is:
f6:61:a8:27:35:cf:4c:6d:13:22:70:cf:4c:c8:a0:23
```

VLAAMS
SUPERCOMPUTER
CENTRUM

Generate the key

Linux/Mac OS users:

- Use *"ssh-keygen " command to generate key pair*
- **Be sure to give your key a passphrase!**
- Default location: ~/.ssh/id_rsa (~/.ssh/id_rsa.pub)
- If other location: `ssh -i location-of-the-file` ......
- **keychain**: ssh agent to load the key with passphrase for current linux session

VLAAMS
SUPERCOMPUTER
CENTRUM
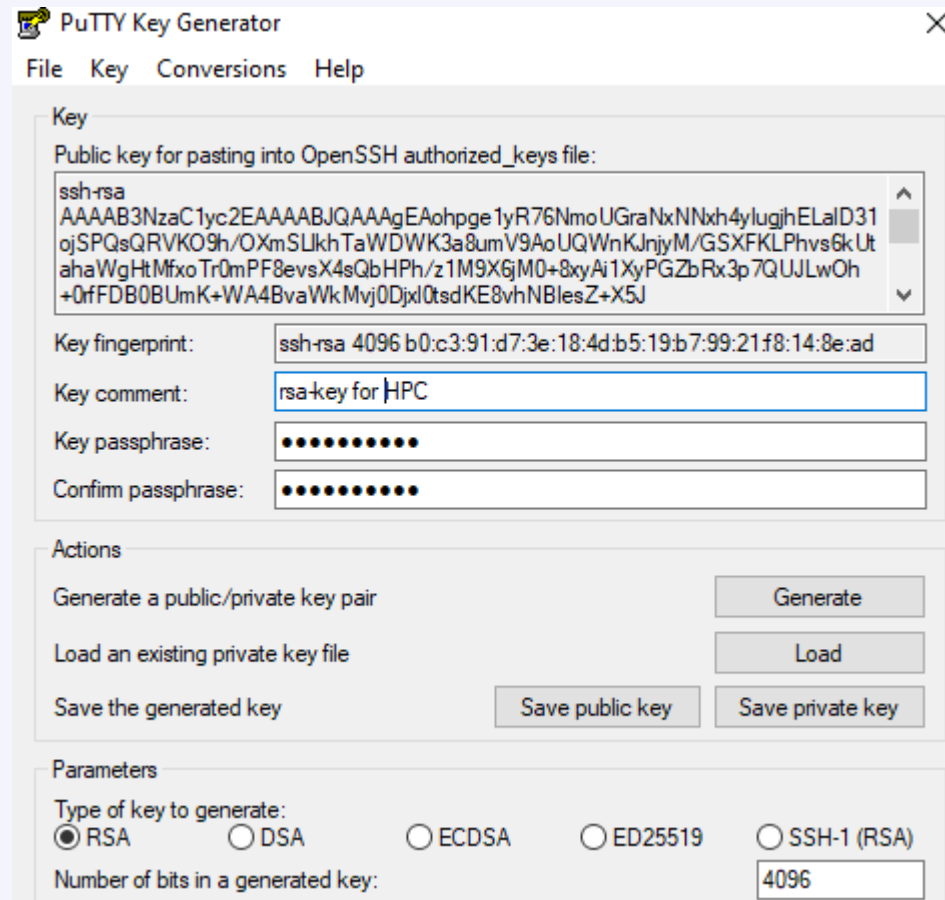
Generate the key

Windows users: use the PuTTYgen key generator.

Request ssh key format: RSA 4096 bit
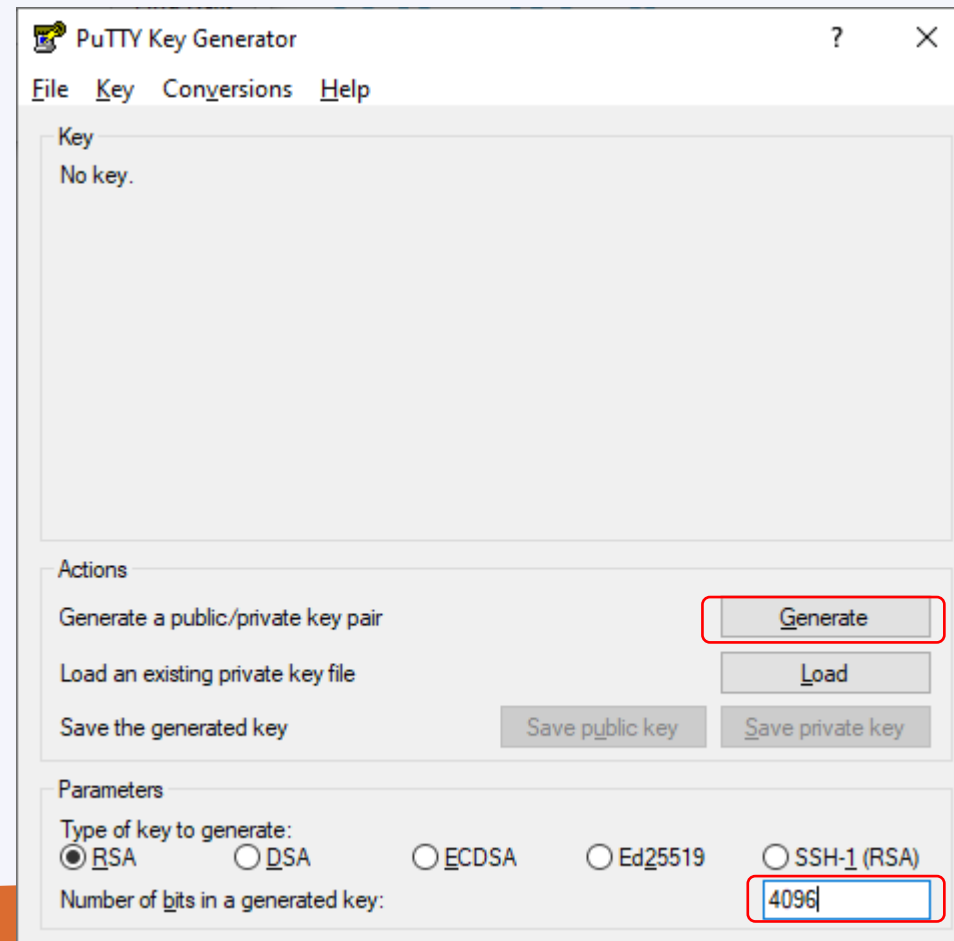
Generate the key

Be sure to give your key a passphrase!

Connecting to the cluster: text mode

Windows users:

- PuTTY is a simple-to-use and freely available GUI SSH client for Windows.

- Pageant can be used to manage active keys for PuTTY, WinSCP and FileZilla so that you don't need to enter the passphrase all the time.
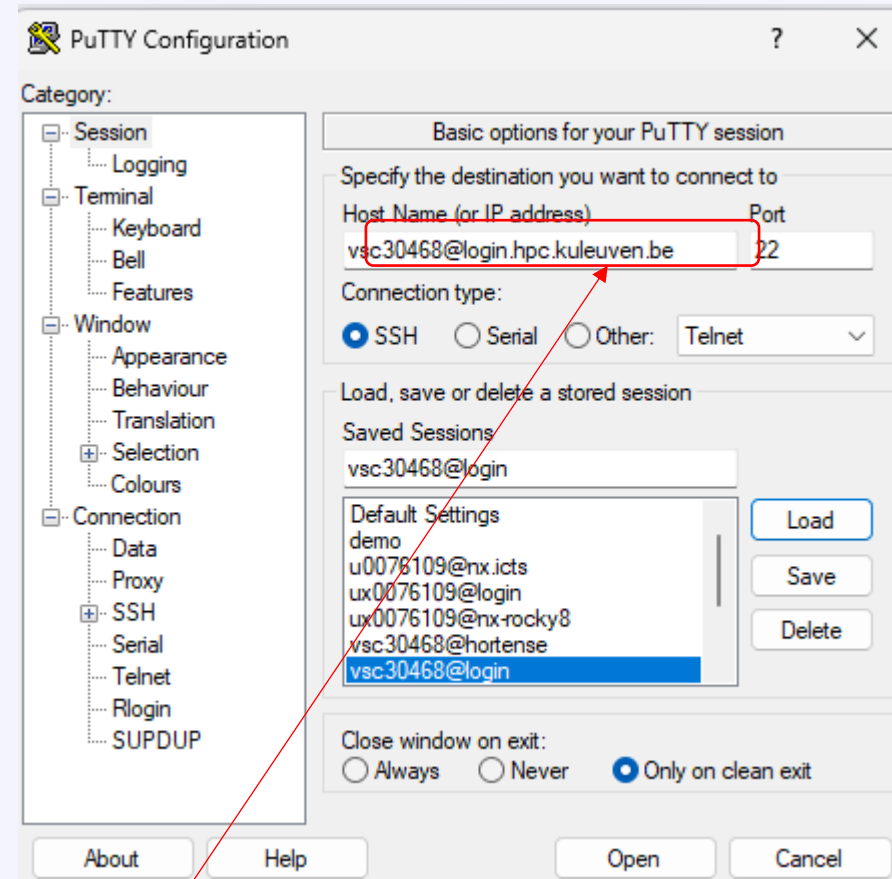
`vsc3XXXX@login.hpc.kuleuven.be`

VLAAMS
SUPERCOMPUTER
CENTRUM

Connecting to the cluster: text mode
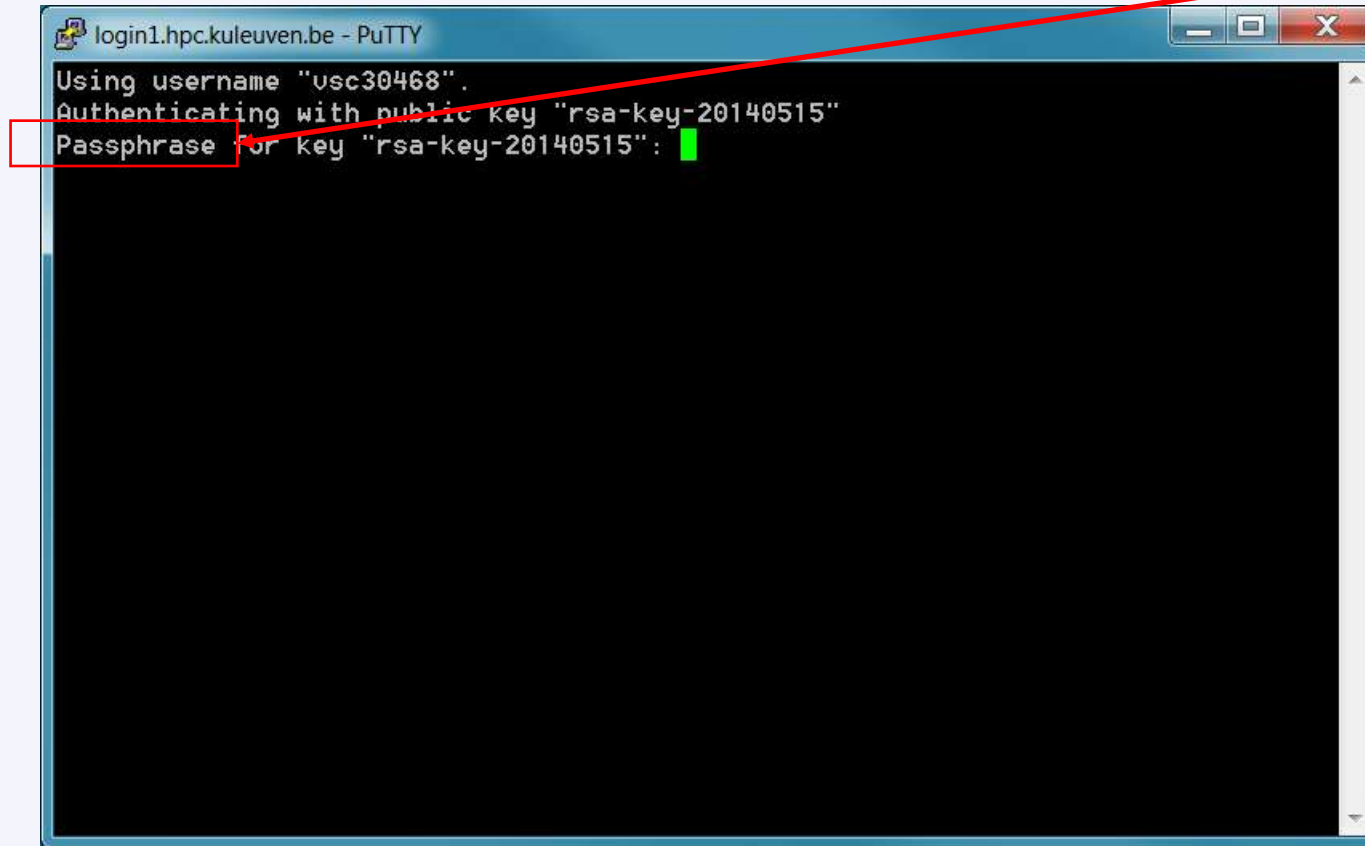
Windows users:

- PuTTY is a simple-to-use and freely available GUI SSH client for Windows.
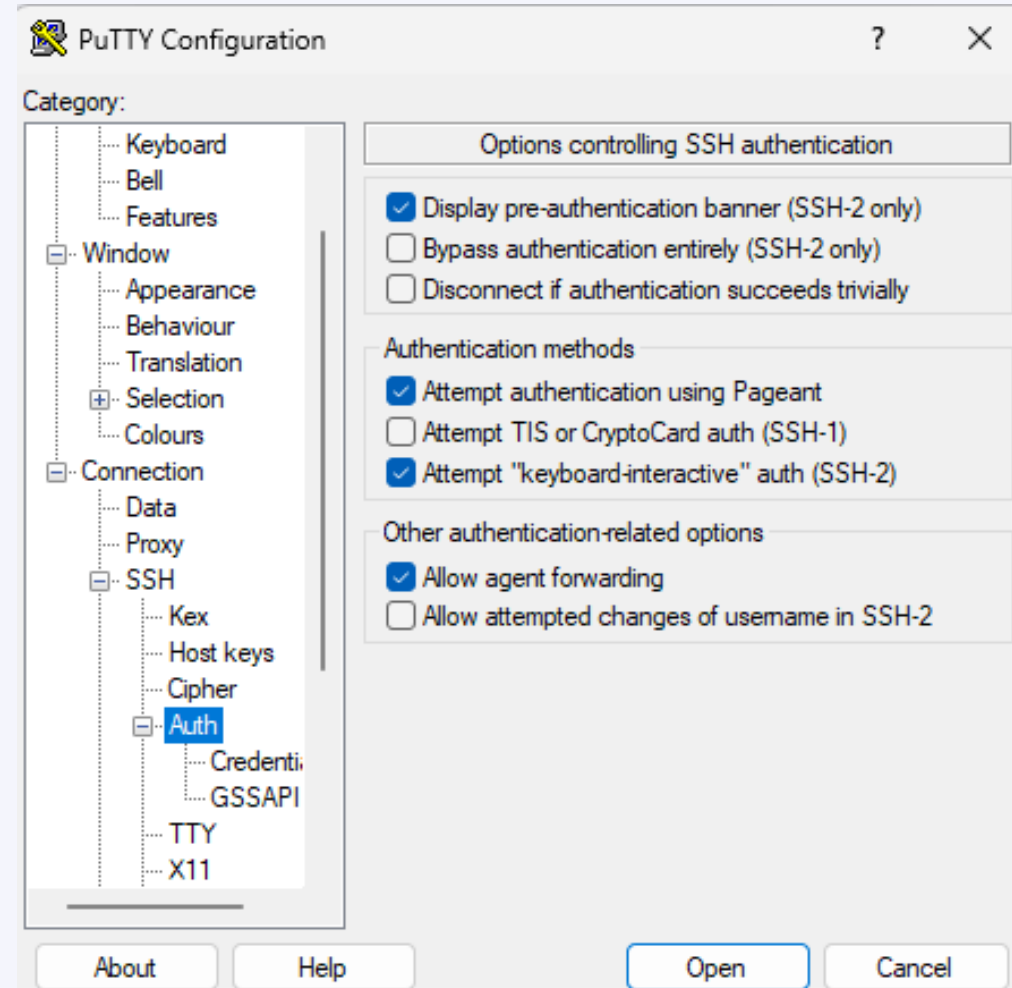
Connecting to the cluster: text mode

Windows users:

If asked for **password** no for passphrase – please stop connecting and contact suport, otherwise after a few attempts you will be blocked for 24hrs
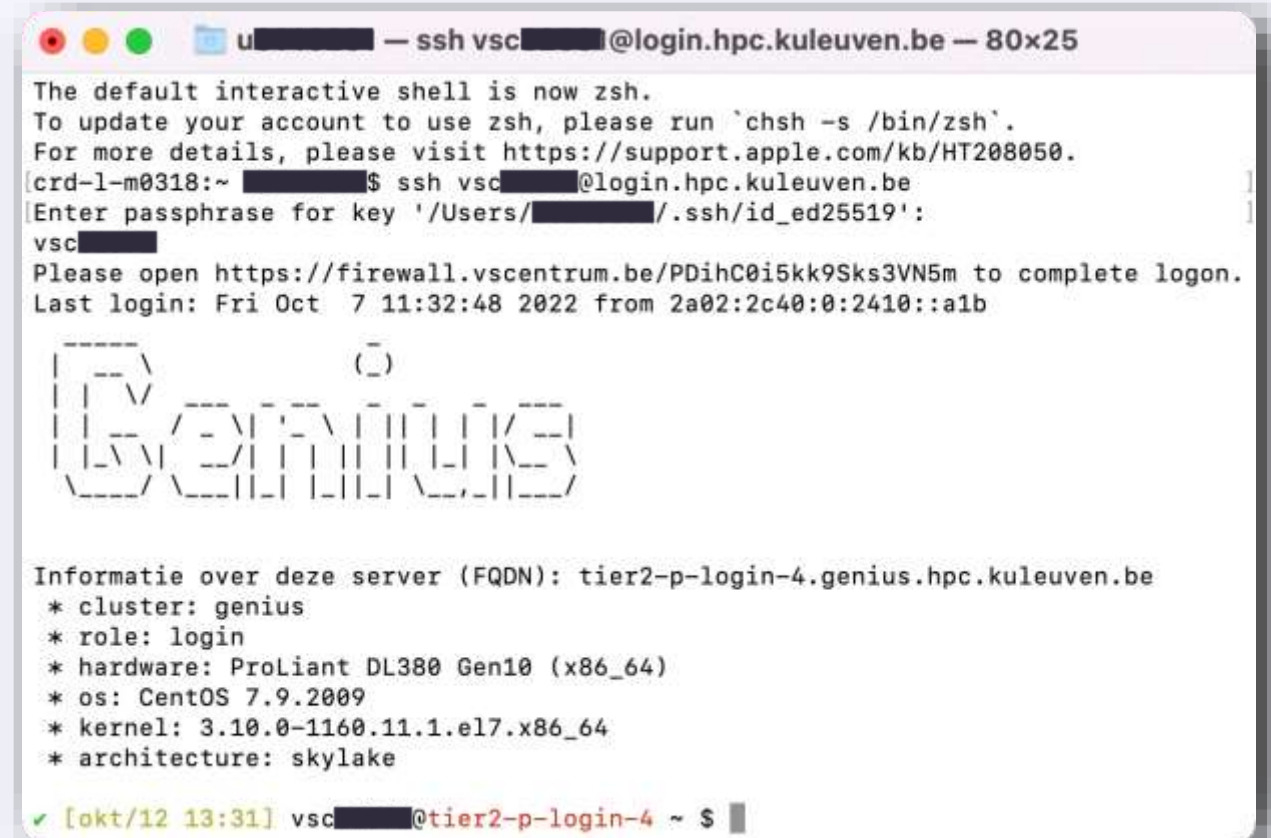
Connecting via Terminal

(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`

```
u_____ — ssh vsc_____@login.hpc.kuleuven.be — 80×25

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 vsc       @login.hpc.kuleuven.be
[Enter passphrase for key '/Users/         /.ssh/id_ed25519':
vsc
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
```

```
                    (_)
  ___        _  _   ___   _   _
 / __| ___  | || | |   | | | | |
| |_ |/ -_) |    | | | | | |_| |
 \___|\___| |_||_| |_|_| \__,_|
```

```
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] vsc       @tier2-p-login-4 ~ $ █
```

© FWO   VScentrum.be

**VLAAMS
SUPERCOMPUTER
CENTRUM**
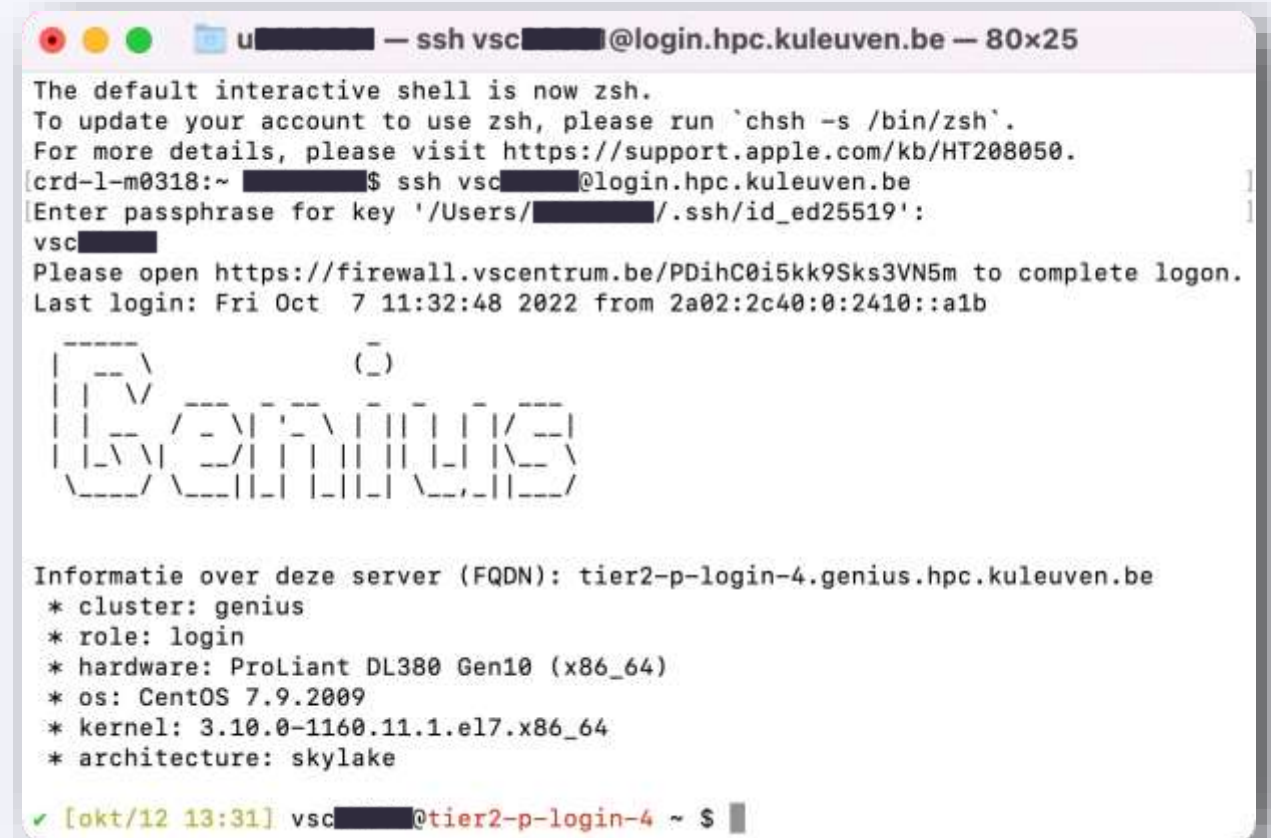
Connecting via Terminal

(Linux and Mac)

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 (MFA):
  `$ ssh vscXXXXX@<hostname>`

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-1-m0318:~          ]$ ssh vsc    @login.hpc.kuleuven.be
[Enter passphrase for key '/Users/        /.ssh/id_ed25519':
vsc
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

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] vsc      @tier2-p-login-4 ~ $
```

VLAAMS
SUPERCOMPUTER
CENTRUM

Connecting to the cluster: display graphics

Windows users:

- PuTTY is a simple-to-use and freely available GUI SSH client for Windows.

- Pageant can be used to manage active keys for PuTTY

- Xming: using X-windows to display graphical programs

Linux users:

- ssh -X vsc3XXXX@login.hpc.kuleuven.be
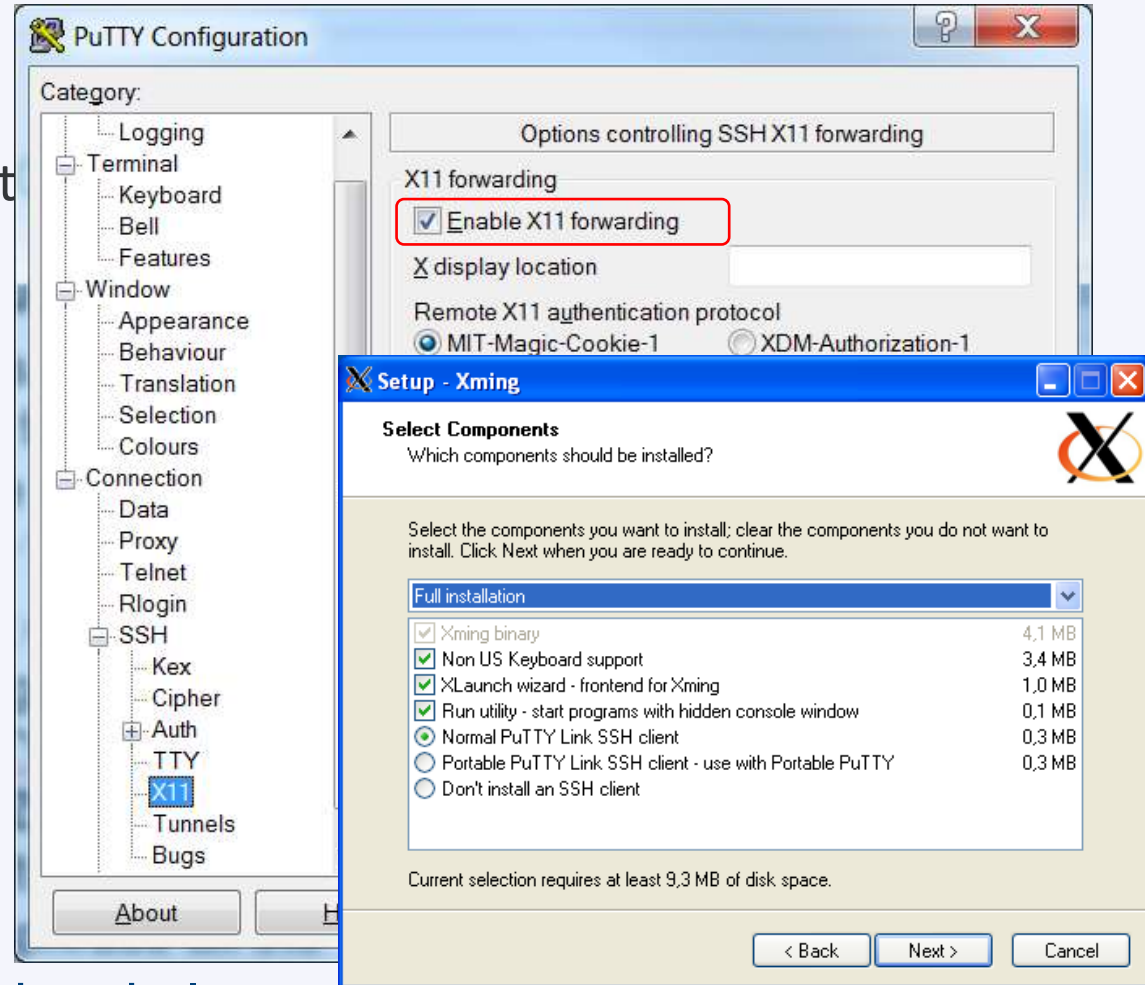
Tier-2 credit system - what

- 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

VLAAMS
SUPERCOMPUTER
CENTRUM

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

Check ICTS service catalogue :
https://icts.kuleuven.be/sc/english/research/HPC
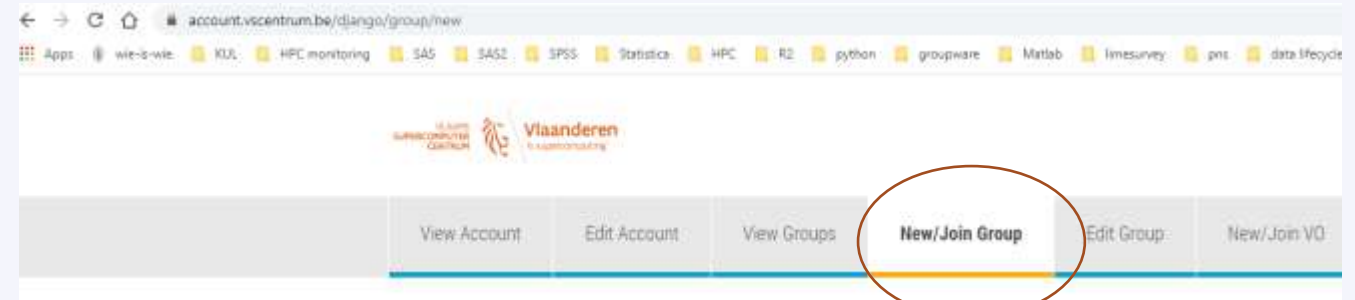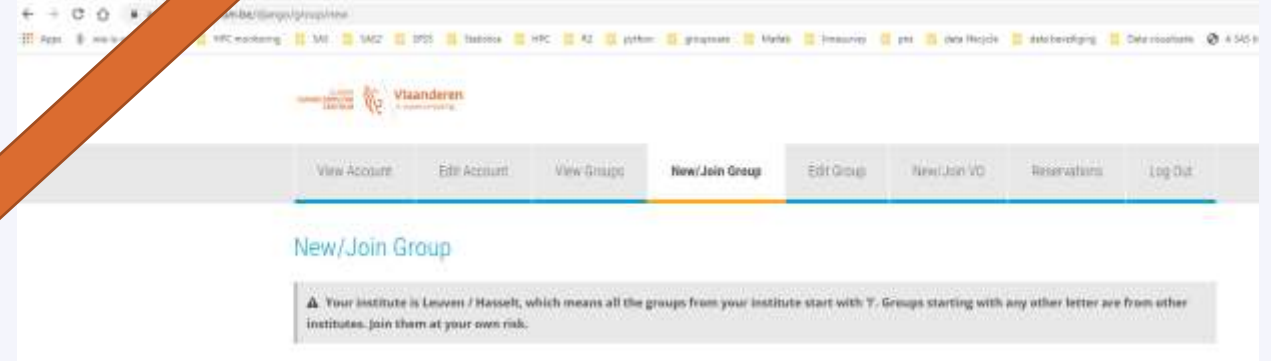
Tier-2 credit system - how

- 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 = minutes * floor(#cpu cores*billing_weight_CPU + #gres*billing_weight_gres)*

  - Taking into account any running jobs

  - When the job finishes, the actual cost is charged

VLAAMS
SUPERCOMPUTER
CENTRUM

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$
```

VLAAMS
SUPERCOMPUTER
CENTRUM

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$
```

VLAAMS
SUPERCOMPUTER
CENTRUM

Tier-2 credit system – manage your credits

- Slurm Accounting Manager => sam- commands

Command	Purpose
`sam-quote sbatch [sbatch arguments] [my_job.slurm]`	Shows the amout of credits a job will use

```
tier2-p-login-3$ sam-quote sbatch --cluster=wice hello.slurm
10980
tier2-p-login-3$
```

VLAAMS
SUPERCOMPUTER
CENTRUM

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$
```

VLAAMS
SUPERCOMPUTER
CENTRUM

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

File Transfer with FileZilla

File transfer
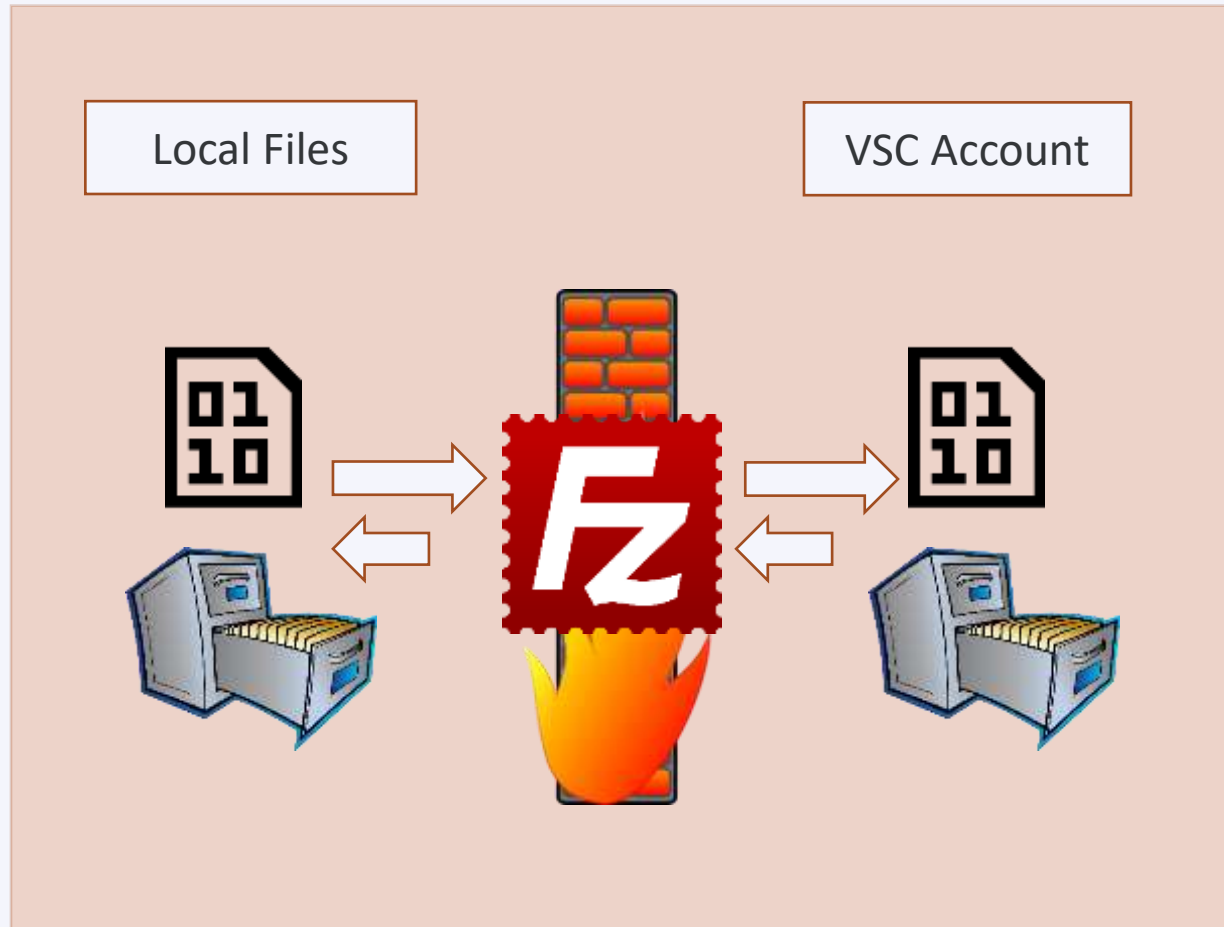
Application	OS
FileZilla	Windows, Linux, Mac
WinSCP	Windows
rsync, scp	Linux, Mac
Globus	Window, Linux, Mac
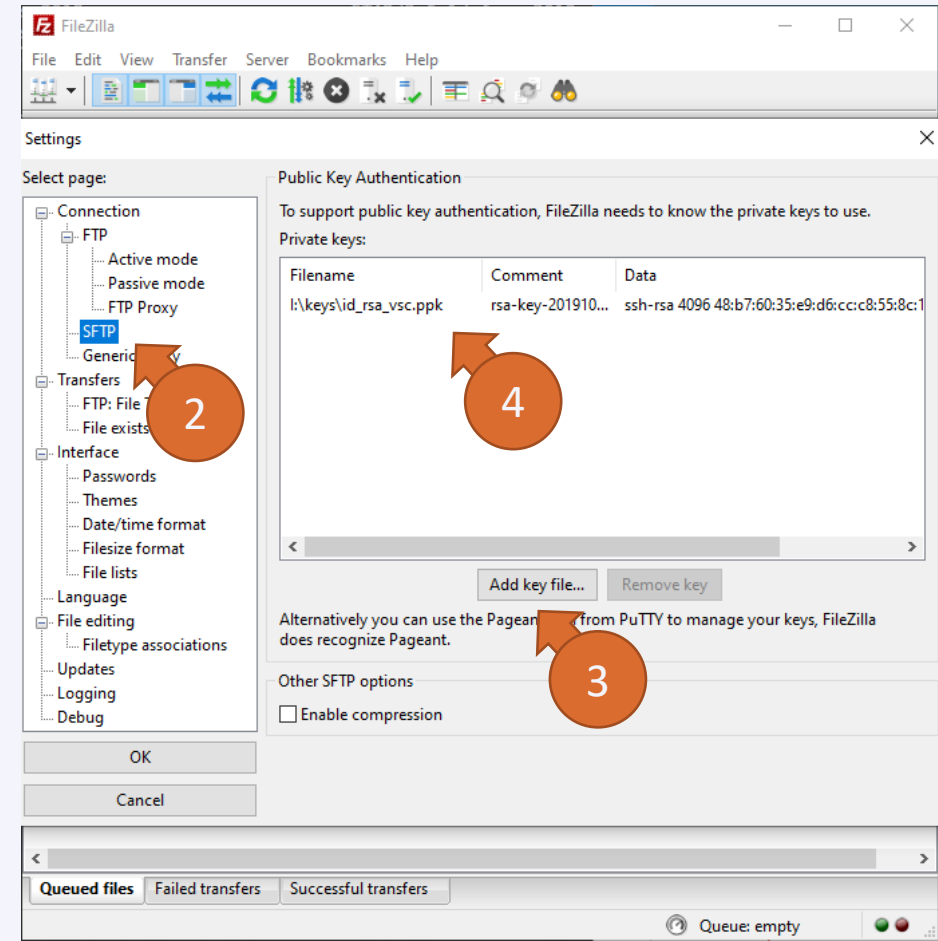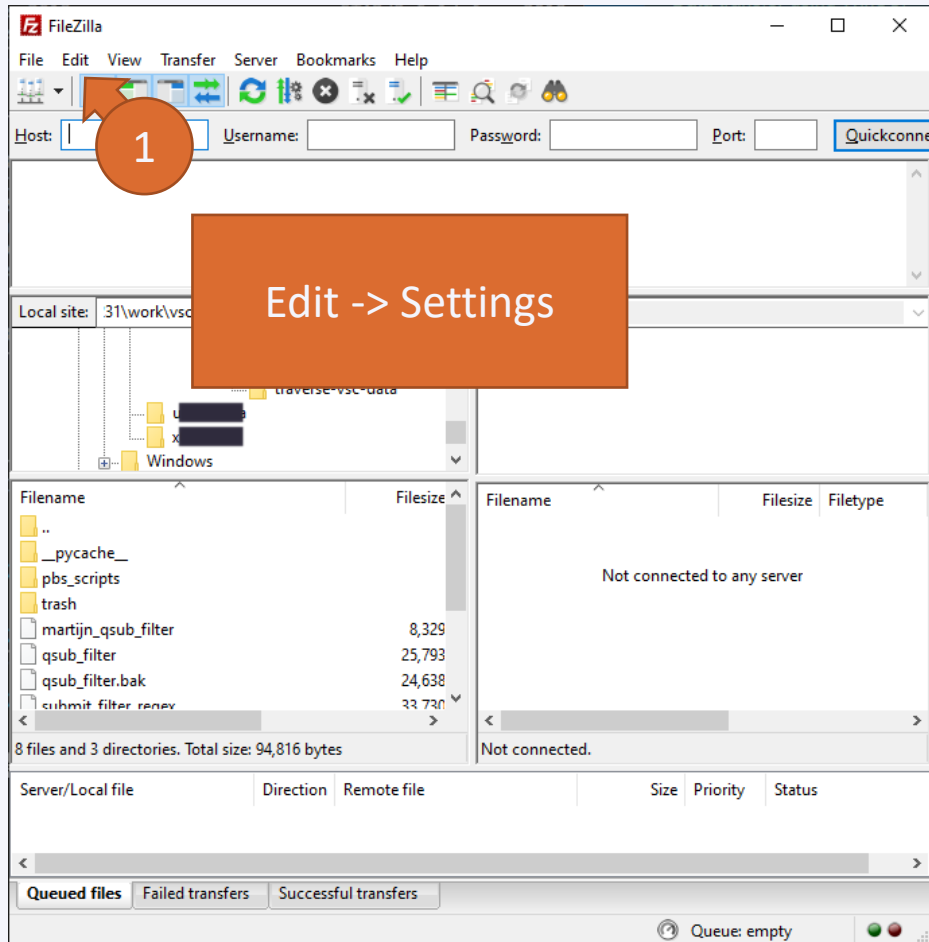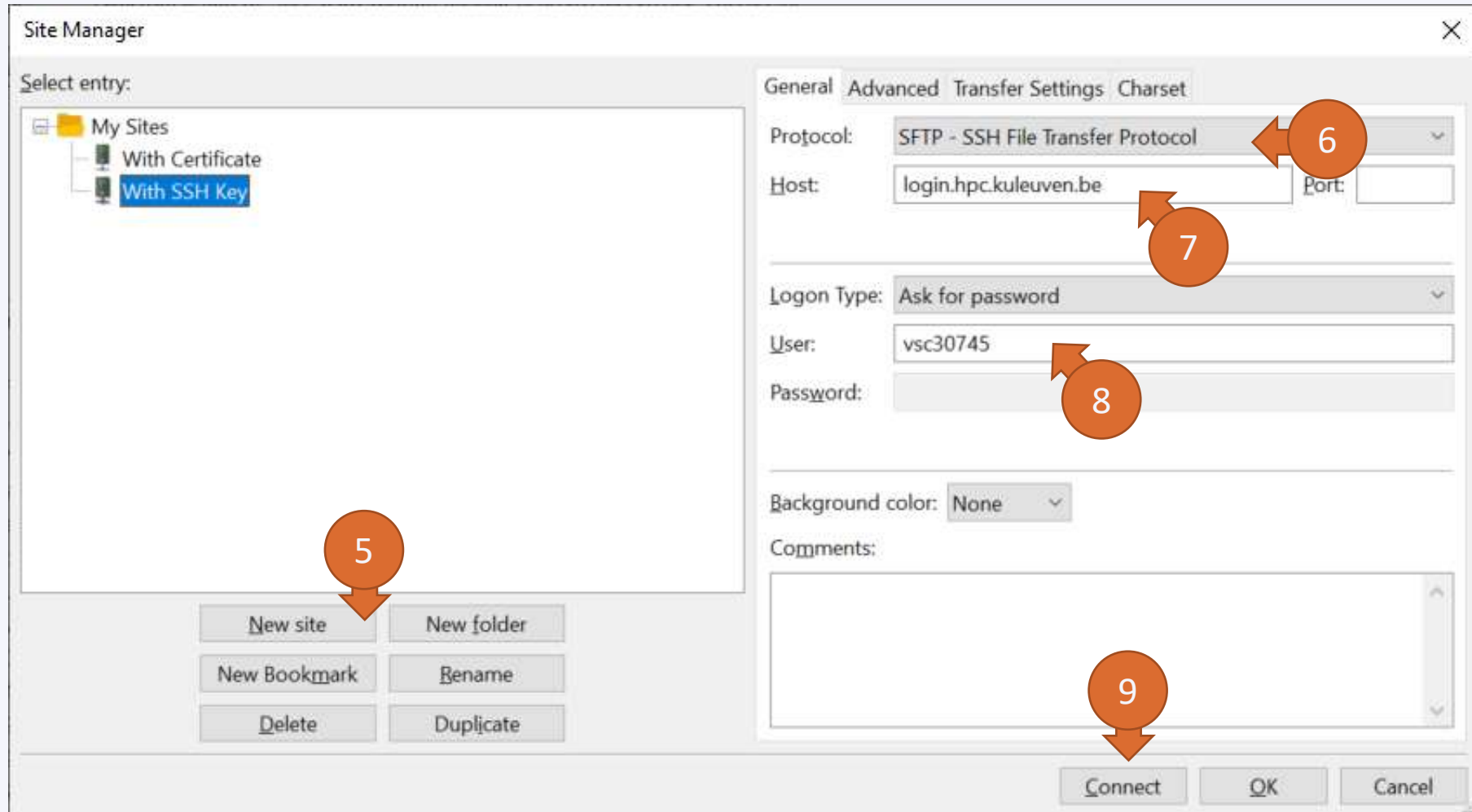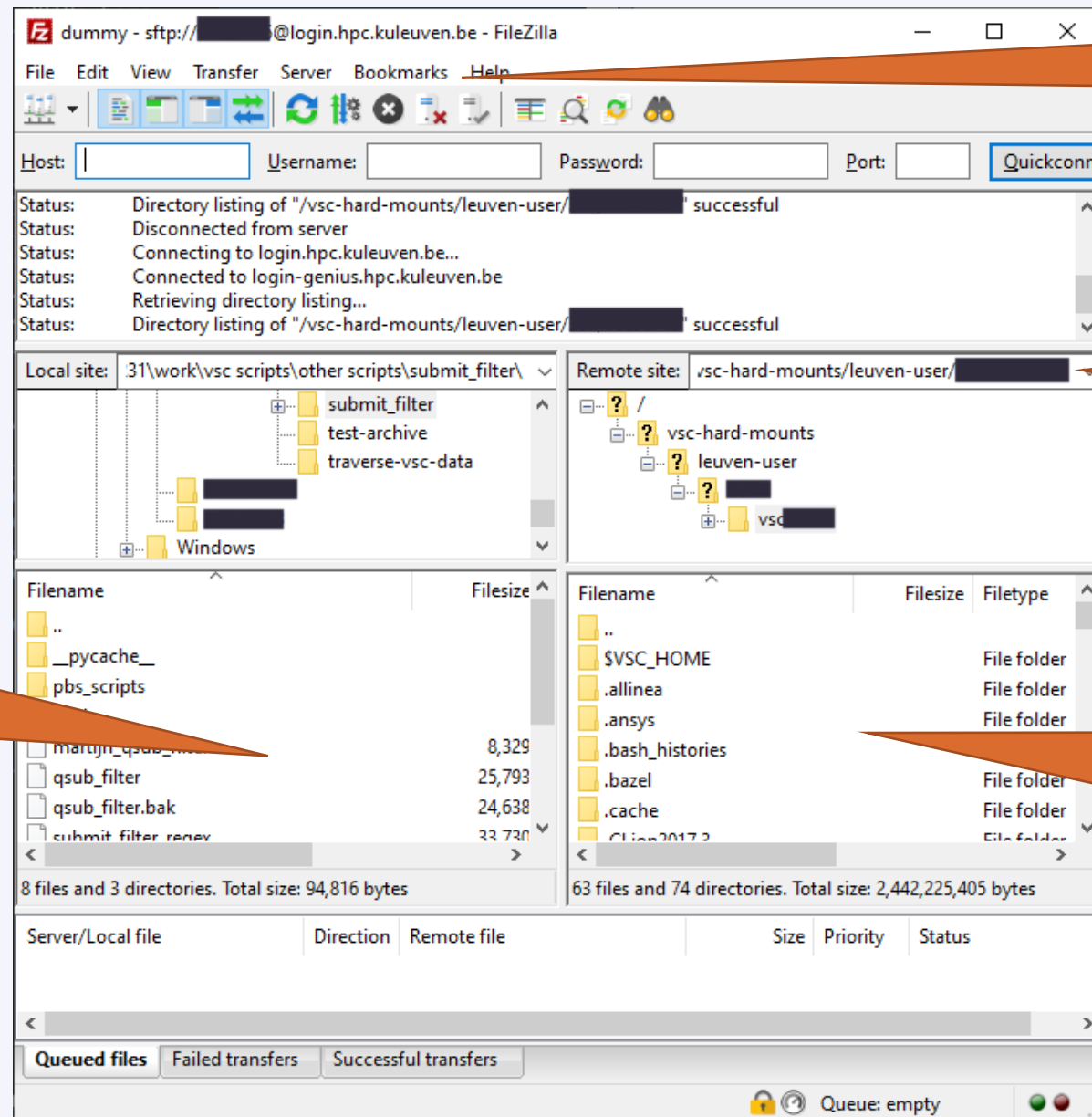
VLAAMS
SUPERCOMPUTER
CENTRUM

File transfer

Edit -> Settings

VLAAMS
SUPERCOMPUTER
CENTRUM

File transfer

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:
  Define "endpoint" on the source
  Define "endpoint" on the destination
  Transfer between endpoints
- ❑ Existing endpoints:
  Tier-2: home/data/scratch storages
  Tier-1: data/scratch
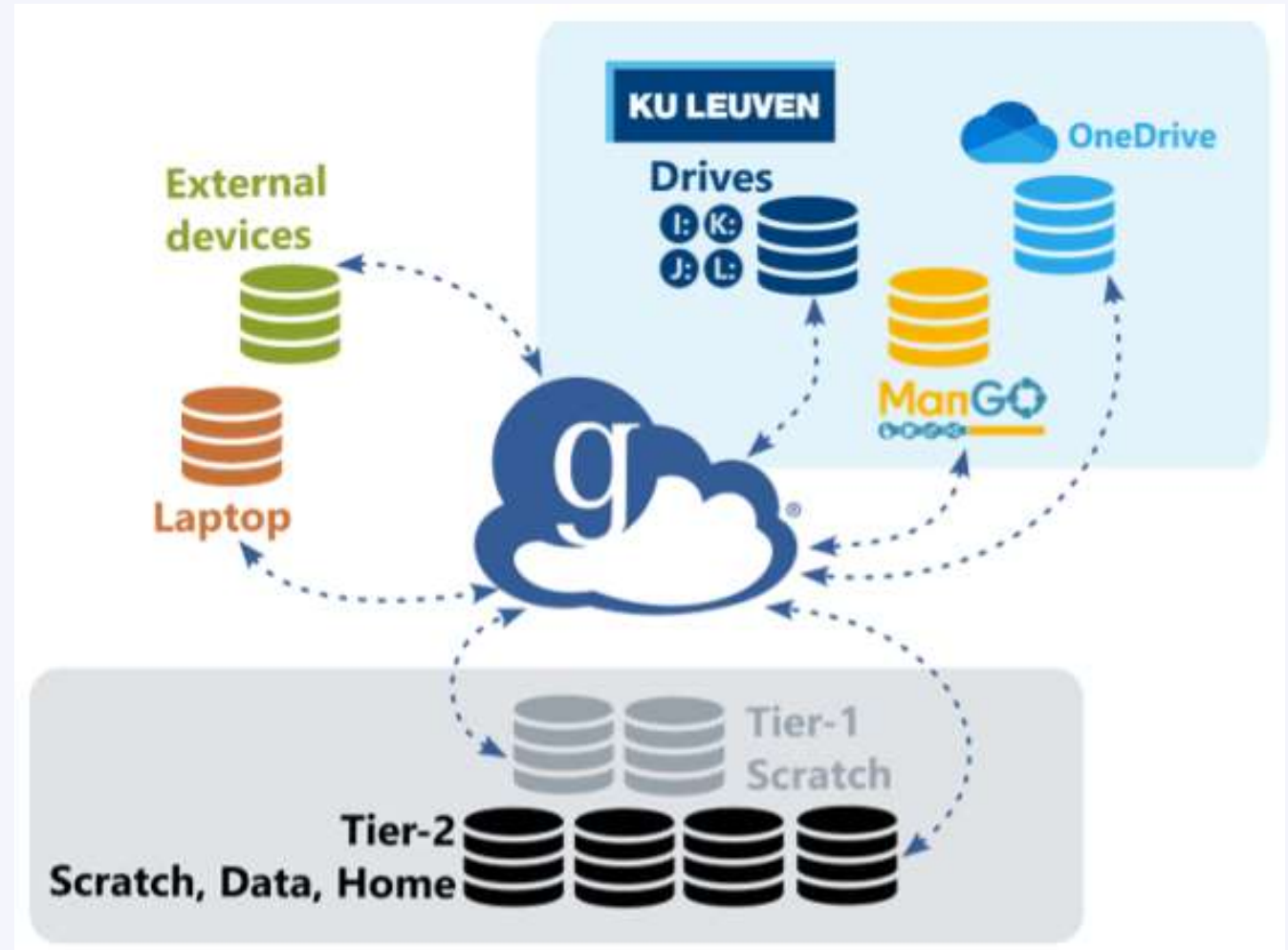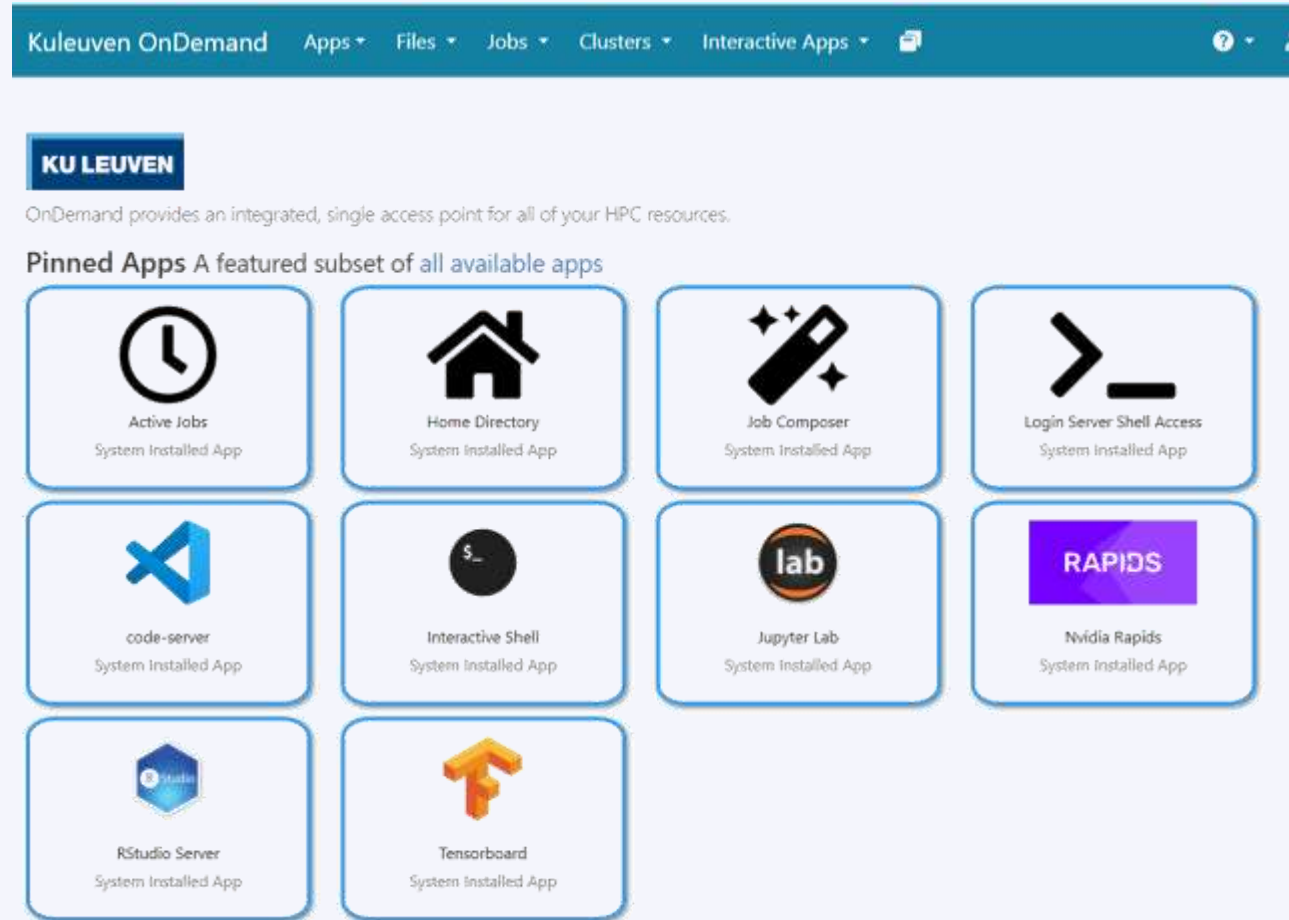
Open Ondemand

Open OnDemand

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

💡

Please close your interactive sessions, when you don't actively use it!

VLAAMS
SUPERCOMPUTER
CENTRUM

Open OnDemand

✅ Use "Files" menu to access your VSC_HOME and VSC_DATA

✅ VSC_SCRATCH is not available

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

🏠 Home Directory
📁 Data Directory   /data/leuven/307/vsc30745
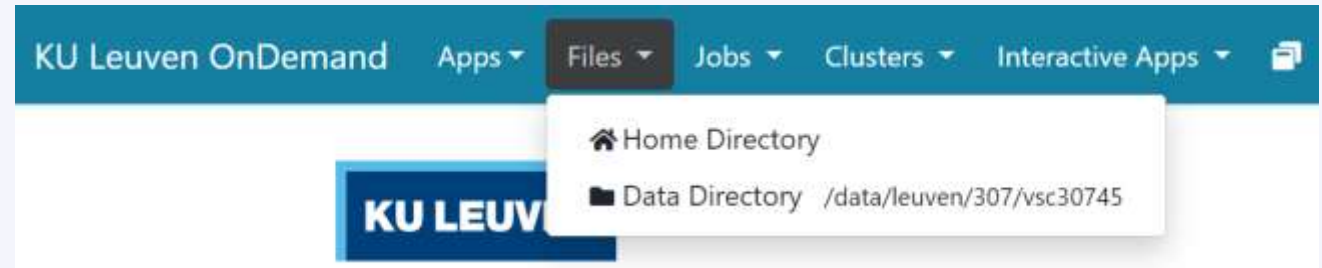
KU LEUV

File/Folder Management

>_ Open in Terminal ▸ | ↻ Refresh | ＋ New File | 🗀 New Directory | ⬆ Upload | ⬇ Download | Globus | 📋 Copy/Move | 🗑 Delete

✅ 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

142

VLAAMS
SUPERCOMPUTER
CENTRUM

Open OnDemand

Monitor/manage all your jobs

Open OnDemand: Login Server Shell Access

Login Shell

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

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

  ___        ___
 |   \       ()
 | |\ \
 | | \ \
 | |  \ \    ___    ___    ___
 | |   \ \  |   |  |   |  |   |  |   |   ___
 |_|    \_\ |___|  |___|  |___|  |___|  |___|
                         Genius

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.vscentrum.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 ~
$
```

VLAAMS
SUPERCOMPUTER
CENTRUM

Open OnDemand: Interactive Shell

Interactive Shell

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

Open OnDemand: Jupyter Lab

Jupyter Lab

- ✅ [Create a notebook](#) as a job (on a compute node)

- ✅ Best practice to:
  pre/post-processing your data
  prototyping
  add figures and text for pedagogical purposes

- ✅ Choose the relevant cluster, partition and resources (core, memory, GPU)

- ✅ Extensions are not supported (for now)

- ✅ User your own Python/R kernels (next slide)

Open OnDemand: Custom R/Python Kernels

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

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 -b -p $VSC_DATA/miniconda3
```

Permanenly add the path to miniconda to your ~/.bashrc
```
$ echo 'export PATH="${VSC_DATA}/miniconda3/bin:${PATH}" ' >> ~/.bashrc
```

VLAAMS
SUPERCOMPUTER
CENTRUM

Open OnDemand: Custom R/Python Kernels

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
```

```
$ python -m ipykernel install -prefix=~/.local --name <env_name>
```
Start a new Jupyter Lab session, and the new kernels must be there

VLAAMS
SUPERCOMPUTER
CENTRUM

Open OnDemand: Other tools

code-server
System Installed App

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.

RStudio Server
System Installed App

Start R IDE as a job.

Install your own packages in $VSC_DATA.

Tensorboard
System Installed App

Track machine learning metrics and visualize data during the workflow.

You need to provide a log folder.

VLAAMS
SUPERCOMPUTER
CENTRUM

Open OnDemand: Resources

E.g. to start a Jupyter Notebook

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

(At the moment) you cannot use scratch and staging

Software Stack

Software: Available Modules

- OS: Rocky Linux 8.x

- Toolchains:

  - Intel (icc, icpc, ifort; Intel MPI; Intel MKL)

  - FOSS (gcc, g++, gfortran; OpenMPI; ScaLAPACK, OpenBLAS, FFTW)

  - Toolchain year on Genius from 2018a; on wICE from 2021a

- Note: **Never mix** FOSS and Intel compilers (gives dependency conflict)

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

VLAAMS
SUPERCOMPUTER
CENTRUM

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

VLAAMS
SUPERCOMPUTER
CENTRUM

Linux File System

Source: http://linuxsuperuser07.blogspot.be/2011/09/rhel-6-file-system.html
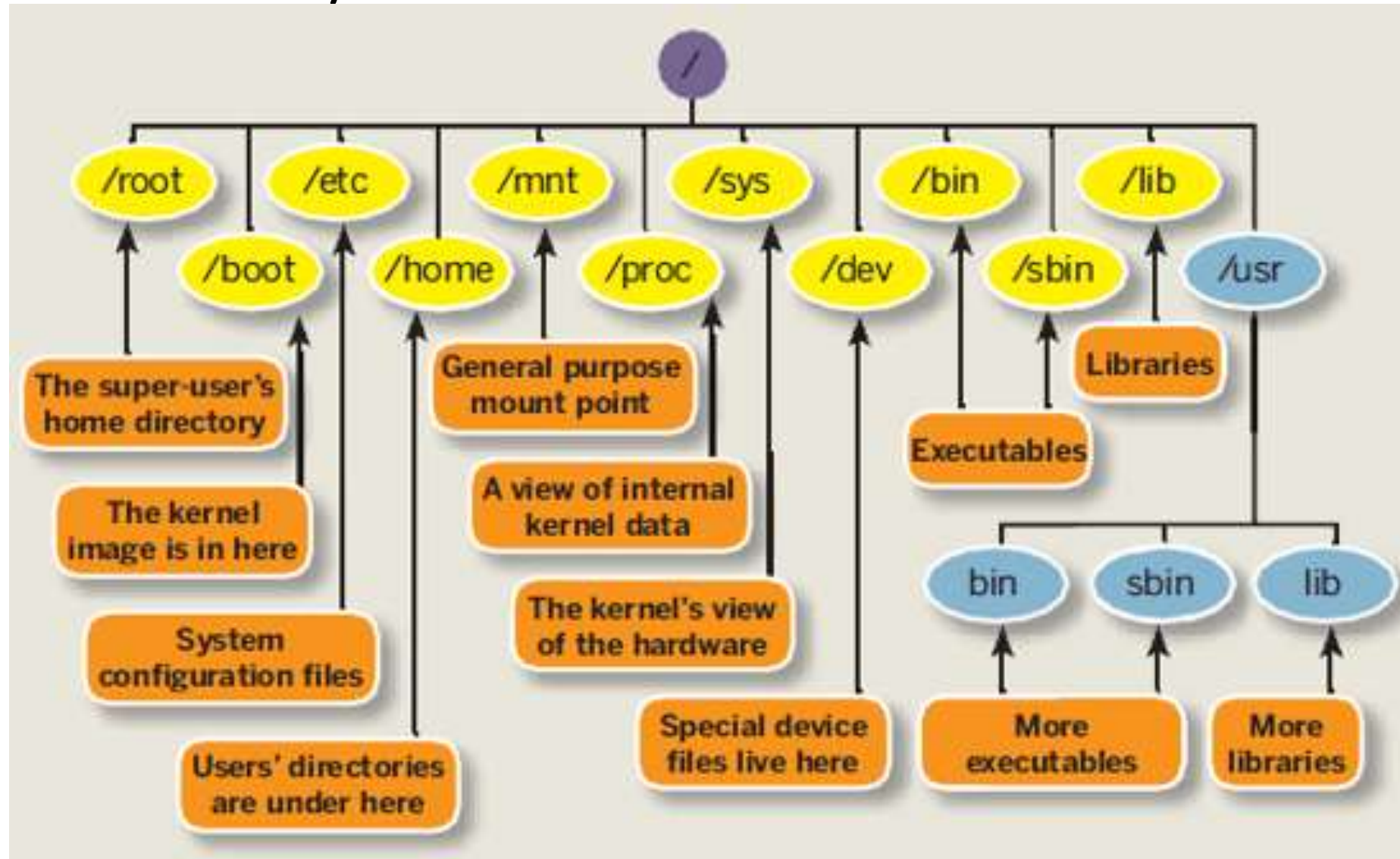
Directory structure
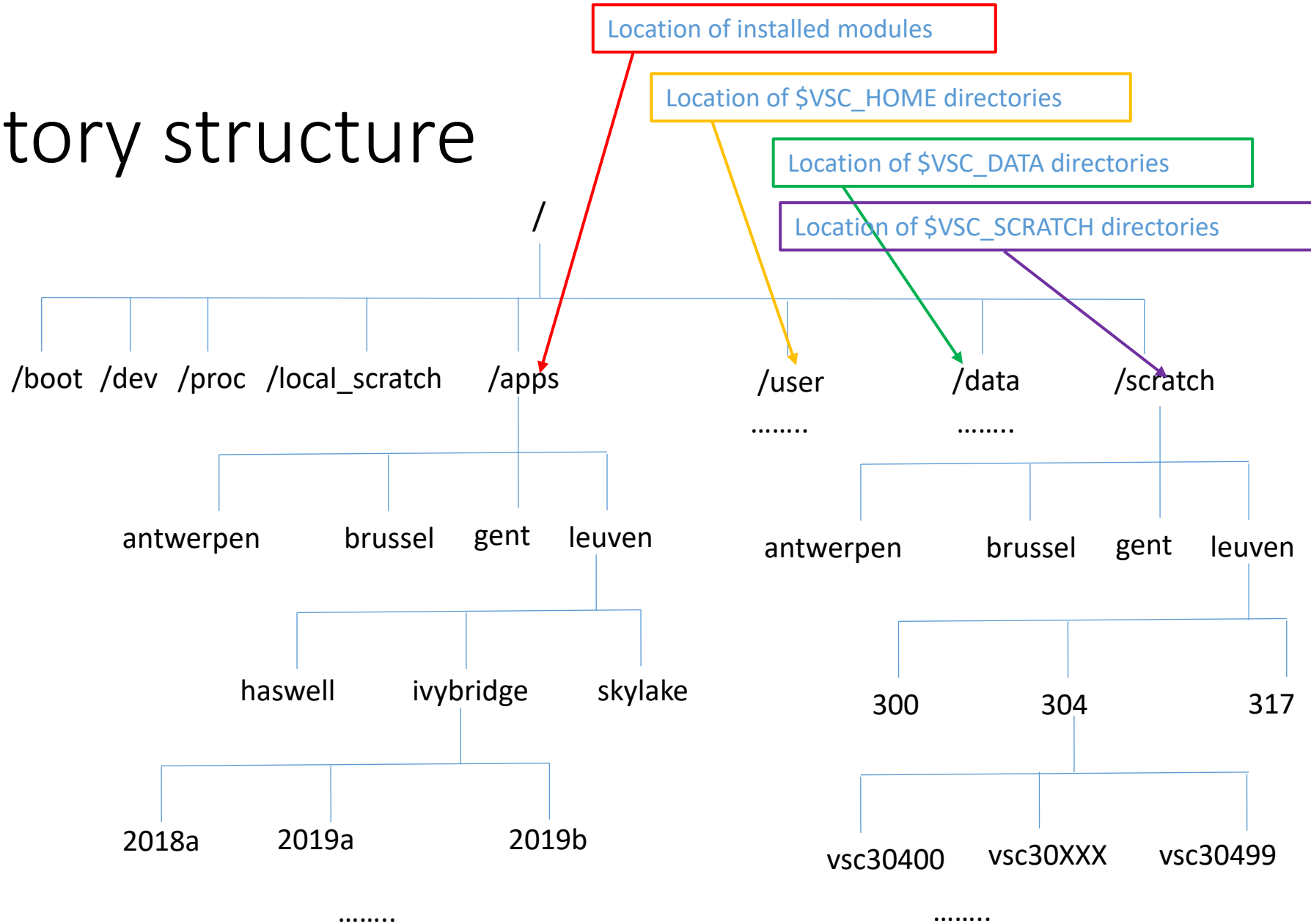
Directory structure
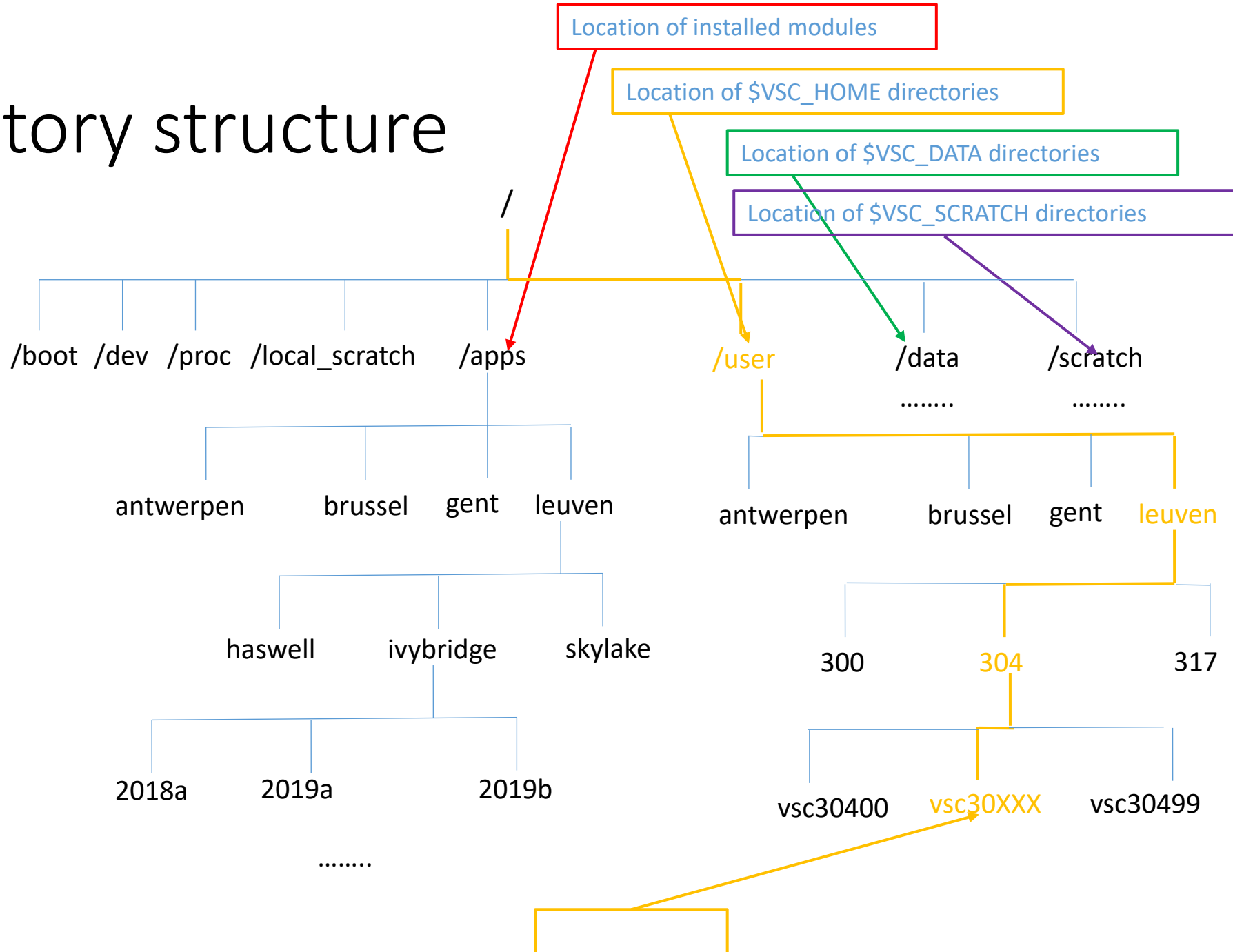
Directory structure
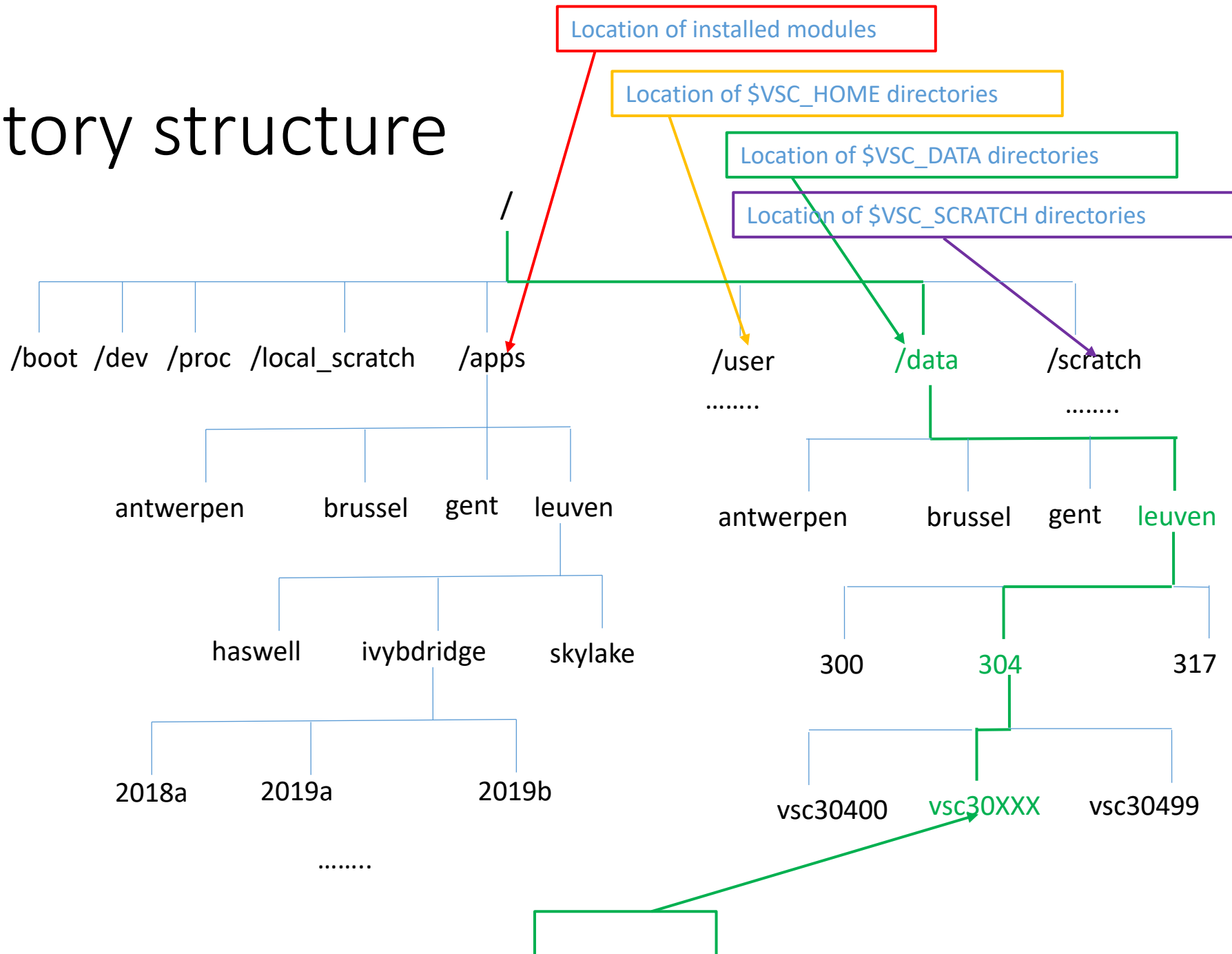
Directory structure
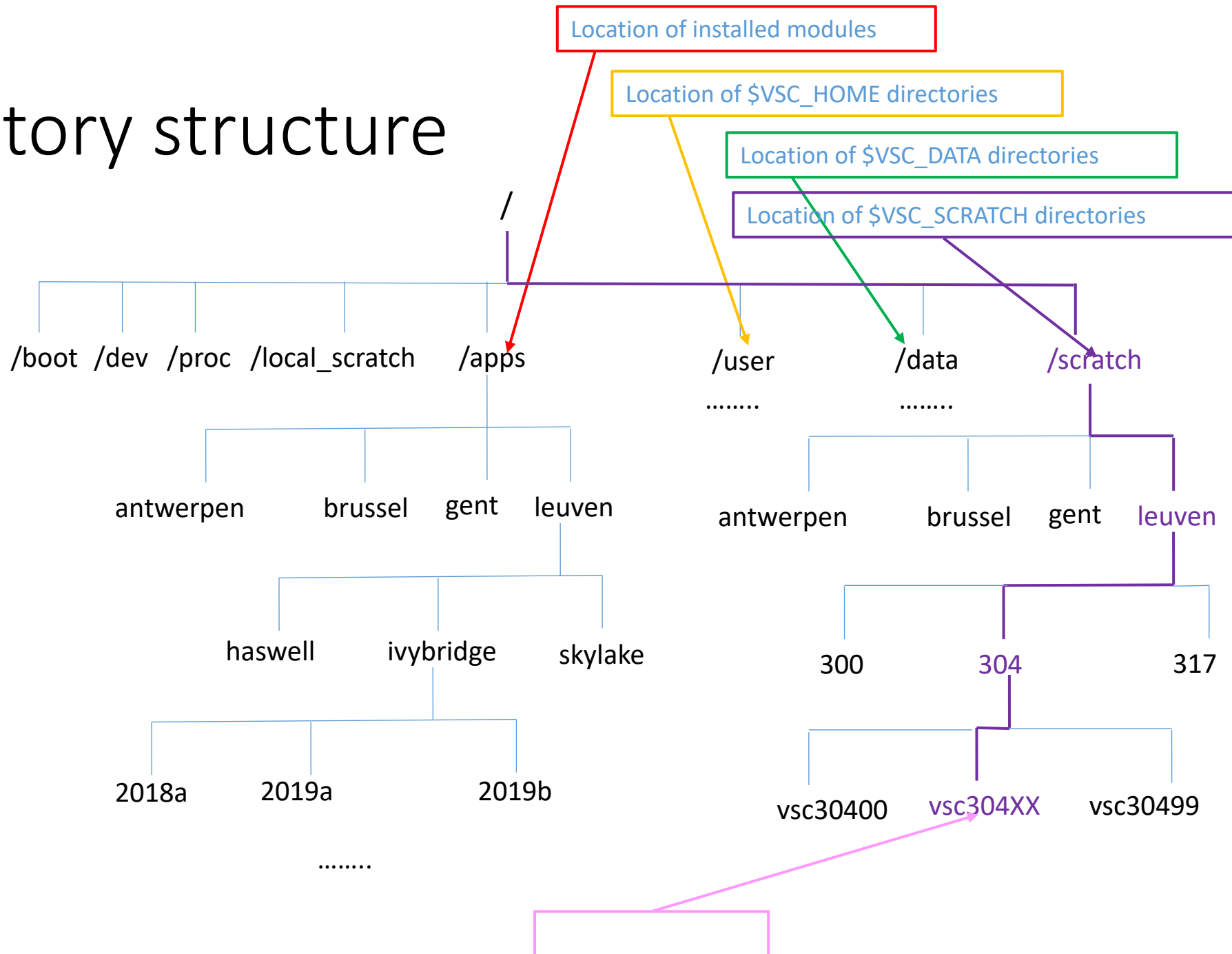
Location of installed modules Genius

Path on Genius:

/apps/leuven/rocky8/skylake/2021a/software/Python/3.9.5-GCCcore-10.3.0
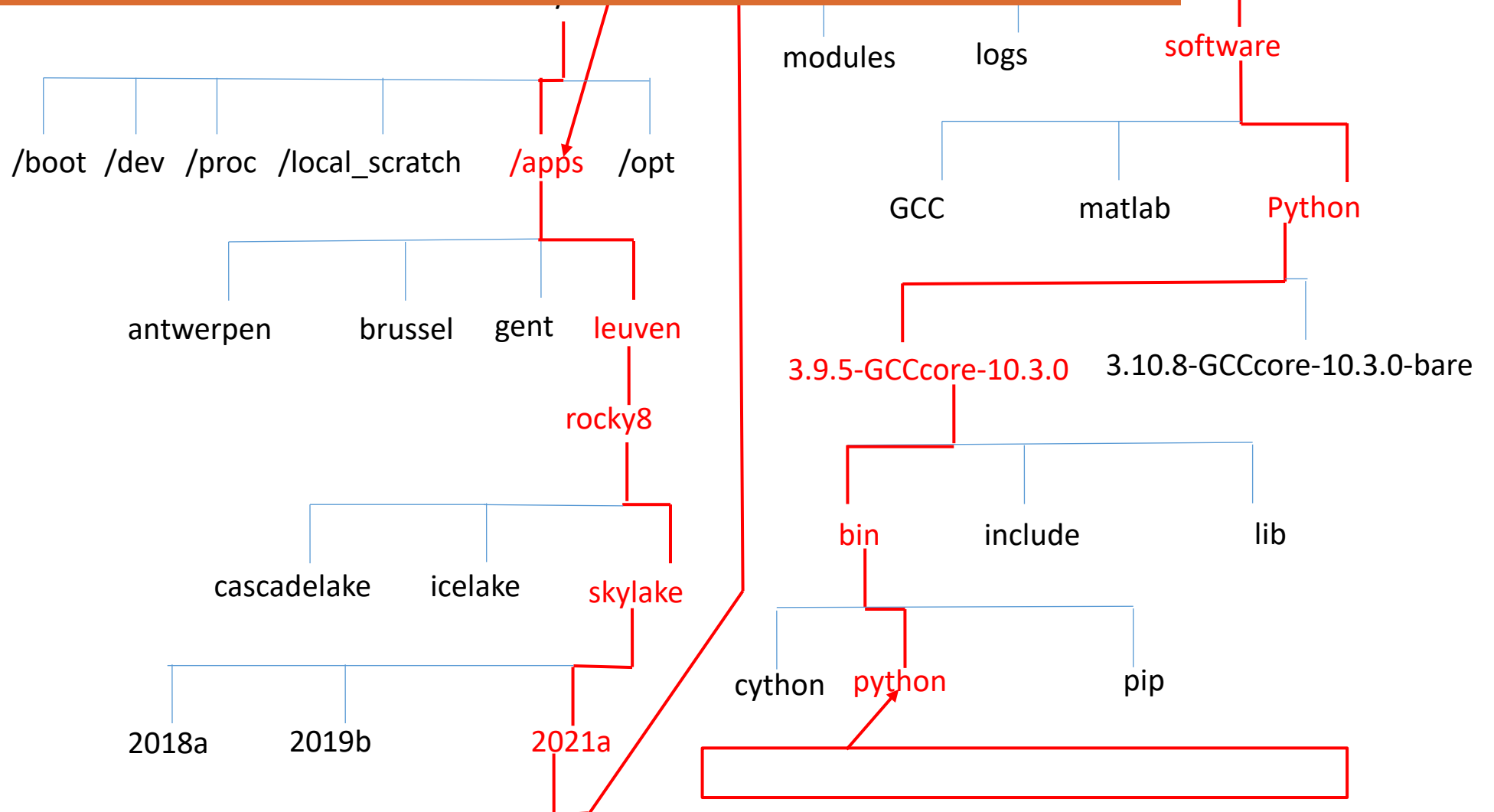
What was I doing???

- Not to loose your job after closing your laptop:
  - Use NX GUI connection
  - Use command line + tmux
    - Start session: `$ tmux new –s test`
    - Detach session: `Ctrl+b+d` (safe to go)
    - List session: `$ tmux ls`
    - Reattach session: `$ tmux a -t test`
    - Exit screen session (within tmux): `$ exit`

What was I doing???

Manage Windows and Session Tabs

Ctrl+B C   Create window

Ctrl+B W   List Windows

Ctrl+B N   Next window

Ctrl+B P   Previous window

Ctrl+B F   Find window

Ctrl+B ,   Name window

Ctrl+B &   Kill window

What was I doing???

Panes/splits

Ctrl+B %	Vertical split
Ctrl+B "	Horizontal split
Ctrl+B O	Swap panes
Ctrl+B Q	Show pane numbers
Ctrl+B X	Kill pane
Ctrl+B Arrow Keys	Move to pane

What was I doing???

- Not to loose your job after closing your laptop:
  - Use command line + screen
    - Start session: `$ screen –S test`
    - Detach session: `Ctrl+a+d` (safe to go)
    - List session: `$ screen -ls`
    - Reattach session: `$ screen –r test`
    - Exit screen session (within screen): `$ exit`

Screen

- Create new window: ctrl-a c
- Go to previous/next window: ctrl-a p/n
- Go to window by number: ctrl-a <window-nr>
- Show current windows, move: ctrl-a ", <window-nr>
- Close window: ctrl-a K
- Detach screen: ctrl-a d
- List current screen sessions: $ screen -ls
- Re-attach to session: $ screen -r <session-id>
- Kill dead session: $ screen -wipe
- Get help: ctrl-a ?
- Monitor for activity: ctrl-a M (same to stop monitoring)
- Monitor for inactivity: ctrl-a _ (same to stop monitoring)

Screen

- Split screen horizontally: ctrl-a S

- Split screen vertically: ctrl-a |

- Go to next screen region: ctrl-a <tab>

- Remove current region: ctrl-a X

- Remove all but current region: ctrl-a Q

- Enter copy mode: ctrl-a [

- Paste: ctrl-a]

- Dump window contents to file: ctrl-a h

- Enable logging: ctrl-a H

- Useful .screenrc file that eliminates some of screen's nuisances:
    - # Turn off that annoying start up message
    - startup_message off
    - # Increase scroll back buffer to a more useful number of lines
    - defscrollback 10000

Screen - settings

- In your **.bashrc** file

```
case ${TERM} in
    xterm)
echo "Hello terminal!!!"
;;
    screen)
echo "Hello screen!!!"
;;
esac
```

Aliasing

- Alias – Alternate name for a command(s)

- You can be inventive with it, but be careful

- If you log out from bash, your aliases will be purged!

- To define an alias permanently, put it in your `.bashrc`

- E.g.
  Image, your jobs always store the results in your scratch folder.
  So, you want an easy way to copy them to your data folder:

  ```
  alias backup="cp -r $VSC_SCRATCH/results
  $VSC_DATA/backup"
  ```

- To disable an alias, do:

  ```
  unalias backup
  ```

Alias and Unalias

- alias newname=oldname
  - eg. *alias copy=cp*
- Then we can use copy in the same way we use cp command
  - eg. *copy file1 file2* //copies content of file1 to file2
- To remove alias use unalias command
  - *unalias copy*
- After this we cannot use copy to perform copying function

Alias

Shells let you define command *aliases*: shortcuts for commands you use very frequently.

Examples

```
alias la='ls -la'
```
Useful to always run commands with default arguments.

```
alias rmi='rm -i'
```
Useful to make rm always ask for confirmation.

```
alias data='cd /data/leuven/304/vsc30XXX'
```
Useful to replace very long and frequent commands.

```
alias schck='. /home/mag/env/chck.sh'
```
Useful to set an environment in a quick way
(. is a shell command to execute the content of a shell script).

which command

Before you run a command, `which` tells you where it is found

```
which ls
alias ls='ls --color=auto'
        /usr/bin/ls

which alias
/usr/bin/alias

which help
/usr/bin/which: no help in
(/usr/local/bin:/usr/local/sbin:/usr/bin:/usr/sbin:/bin
:/sbin:/home/mag/.local/bin:/home/mag/bin)
```

~/.bashrc file

`~/.bashrc`
Shell script read each time a bash shell is started (login, or when the job starts on a compute node)

You can use this file to define

- Your default environment variables (PATH, EDITOR...).

- Your aliases.

- Your prompt (see the bash manual for details).

- A greeting message.

Do NOT put "`module load`" in your .bashrc. It creates conflicts

bash configuration Files

- bash has two different login files.
  - **.bashrc** gets read when you open a local shell on a machine
  - **.bash_profile** only gets read if and only if you login from a remote machine. Note that **.bash_profile** itself reads in your **.bashrc** file as well.

- If you want aliases to be executed regardless, then you should put them in the **.bashrc** file.

- On the cluster please edit only .bashrc file – in case of problem we can always allow you access thanks to correct .bash_profile

~/.bash_profile

- This is how your ~/.bash_profile looks like

- Tip: never touch it

~/.bash_profile

```
# File:  .bash_profile
# Get the aliases and functions
# Get whatever is in your
# .bashrc config file
if [ -f ~/.bashrc ]; then
        . ~/.bashrc
fi
```

```
# File: .bashrc #
# Description: A default .bashrc
###Source global defs ###
if [ -f /etc/bashrc ]; then
    . /etc/bashrc
fi

###set the prompt ###
# uncomment out only one
# this is hostname and time
PS1="\h-(\@): "
# this is hostname and
# history number
#PS1="\h-(\!)# "
# this is hostname and
# working directory
#PS1="\h-(\w)# "
# this is hostname and
# shortened working
# directory
#PS1="\h-(\W)# "
```

```
### path manipulation ###
# add ~/bin to the path,
# cwd as well
PATH="$PATH:$HOME/bin:./“

### env variables ###
# make sure that you
# change this to your
# username
MAIL="/afs/umbc.edu/users/u/s/username/M
    ail/inbox“
export PATH
unset USERNAME

### User-specific aliases
### and functions ###
alias rm="rm -i"
```

Environment Variables

- Use the env command to see all environment variables
- set/export to see all shell variables
- Set or change environment variables from the command-line:
  new values last only for current login session.

sh/bash/ksh   set: NEW_VARIABLE=newvalue
            append: OLD_VARIABLE=$OLD_VARIABLEnewvalue
            prepend: OLD_VARIABLE=newvalue$OLD_VARIABLE
                add: OLD_VARIABLE=${OLD_VARIABLE}:newvalue
            export OLD_VARIABLE

The order decides where system checks for command first (important if you have your own version and there is another vesion on the cluster)

Introduction to bash

- The bash shell is one of the many shells that are available to you on the VSC nodes.

- Almost any installation of Linux defaults to the bash shell.

- bash is one of the many GNU.org (http://www.gnu.org) projects.

- bash manuals:
  - A comprehensive online manual is provided at http://www.gnu.org/software/bash/manual/bashref.html
  - Aliases - http://www.gnu.org/software/bash/manual/bashref.html#Aliases
  - Controlling the Prompt - http://www.gnu.org/software/bash/manual/bashref.html#Controlling-the-Prompt

Universal customization

- Universal .bashrc - written to run on all (relevant) clusters:

```
case ${VSC_INSTITUTE_CLUSTER} in
    wice)
        ulimit -c 500000
        export LD_LIBRARY_PATH="${HOME}/lib:${LD_LIBRARY_PATH}"
        export PATH="${HOME}/bin:${HOME}/sbin:${PATH}"
        export EDITOR="/usr/bin/vim"
        export PS1=': \u@\[\e[1;31m\]\h\[\e[0m\] \w `date
+%H:%M` $'
        source ${HOME}/.autoenv/activate.sh
        ;;
    genius)
        export EDITOR="/usr/bin/vim"
        alias vim="vim -u .vimrc-simple"
        export PS1=': \u@\[\e[1;34m\]\h\[\e[0m\] \w `date
+%H:%M` $ '
        ;;
esac
```

Starting to Compute

Resource Glossary

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

VLAAMS
SUPERCOMPUTER
CENTRUM

Default Resources

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

VLAAMS
SUPERCOMPUTER
CENTRUM
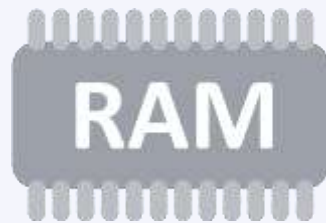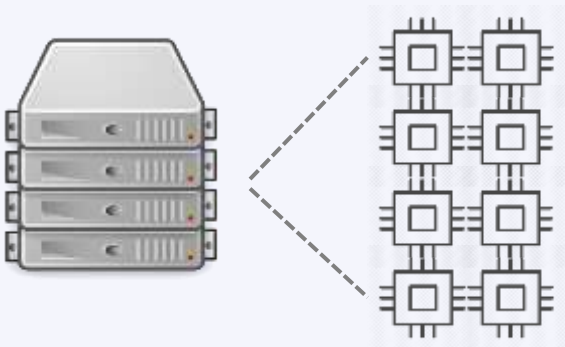
Serial Application
(1 process on 1 core)

Single Process
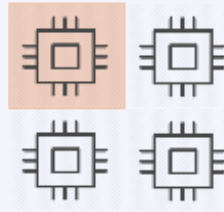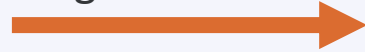
One node

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

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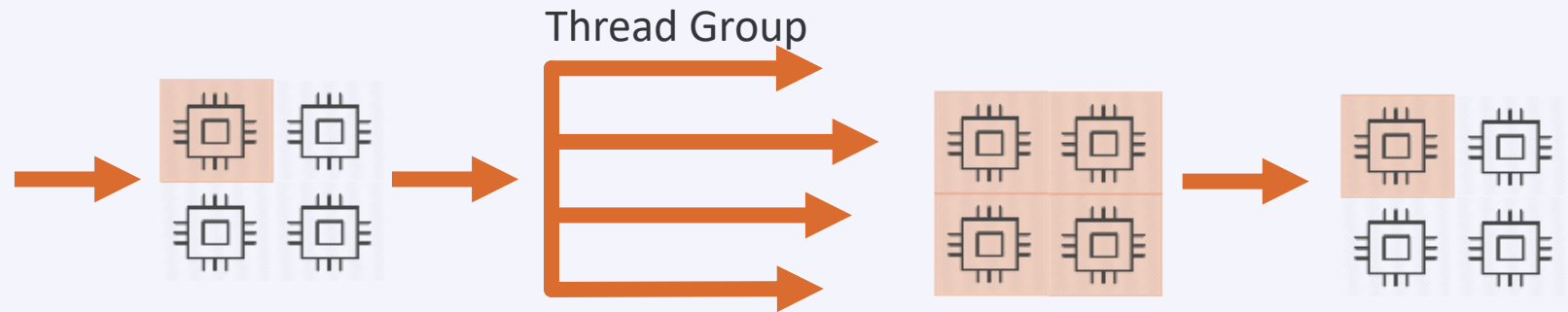
Distributed Appl.
(many processes on multiple cores/nodes)

Process Group

VLAAMS
SUPERCOMPUTER
CENTRUM

Backfill

Nodes

cores ↑ → time

Job queue

Job 1 | Job 2 | Job 3 | Job 4 | Job 5 | Job 6

Interactive Job

Login node

srun …

Compute nodes

Batch Job

Login node

sbatch script.slurm

Compute nodes

VLAAMS
SUPERCOMPUTER
CENTRUM

Interactive Job on **wICE**

Request interactive job

Project Name for Compute Credits

For special hardware or queue

Target cluster

```
$ srun --account=<project name> --partition=interactive --clusters=wice \
```

```
--nodes=1 --ntasks=4 --time=01:00:00 --pty bash -l
```

Number of nodes

Number of processes or cores

Max. Runtime

Interpret with Bash

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`

VLAAMS
SUPERCOMPUTER
CENTRUM

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

VLAAMS
SUPERCOMPUTER
CENTRUM

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; defaullt="slurm-%j.out"
`--error=<file_template>`	Default: redirect to STDOUT
`--mail-type=FAIL,BEGIN,END`	
`--mail-user=<email-address>`	
`--export=<ALL,key=value>`	Additional variables to pass to job
`--account=<account_name>`	Mandatory (credits)

VLAAMS
SUPERCOMPUTER
CENTRUM

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

VLAAMS
SUPERCOMPUTER
CENTRUM

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
```
$ sattach <JobID>.<StepID>   # e.g. 55439251.0
```

VLAAMS
SUPERCOMPUTER
CENTRUM

Example Slurm Job Script

jobscript.slurm

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

Shebang

Resource List

Module load(s)

Move data

Execute commands

Move data

VLAAMS
SUPERCOMPUTER
CENTRUM

Batch Workload

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

```
#!/bin/bash -l
…
```

```
$ sbatch simulation.slurm
```

Submitted batch job 60042478 on cluster wice

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

VLAAMS
SUPERCOMPUTER
CENTRUM

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
```

VLAAMS
SUPERCOMPUTER
CENTRUM

Output File

- Always created

- `slurm-<job_id>.out`

- Contains all standard output and error (instead of screen)

- Always study it

- Standard Output and Error channels can be redirected to other files:
  `#SBATCH --output …`
  `#SBATCH --error …`

stdout

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

Output File

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

Resource Summary

Stdout

VLAAMS
SUPERCOMPUTER
CENTRUM

Output File

```
$ 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

VLAAMS
SUPERCOMPUTER
CENTRUM

Interactive Partitions

- ✅ 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
- ✅ 12 nodes on Genius: 36 cores/node, 192 GB mem
- ✅ 4 nodes on wICE: 64 cores/node, 512 GB mem, 1 GPU (=7 GPU instances)
- ✅ 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
```

Debugging Partitions

- Accessible via command line and Open OnDemand
- Quickly test if your (parallel) application works
- Short queue time
- Only one job at a time, max walltime: 1 hr

Cluster	Partition	Resources
Genius	batch_debug	2x Cascadelake
	gpu_p100_debug	1x Skylake node (4x P100 GPUs)
wICE	interactive	4x Icelake (1x A100 GPU)
	gpu_a100_debug	1x Icelake (4x A100 GPUs)

Demo: Test yourself

Basics

- Request membership to lp_hpcinfo group (account.vscentrum.be)

- Login via OpenOnDemand: www.ondemand.hpc.kuleuven.be
  Features: File transfer, File editor, interactive apps, Jobs' overview

- Check disk quota:
  ```
  $ myquota
  ```

- Check the credits
  ```
  $ sam-balance
  ```
  and
  ```
  $ sam-list-allocations
  ```

- Software modules:
  ```
  $ module {avail|list|load|unload|purge}
  ```

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
- While the job is running get the information about the node `slurm_jobinfo...`

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`)

VLAAMS
SUPERCOMPUTER
CENTRUM

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 --cluster=wice <JobID>` `$ slurm_jobinfo <JobID>`	Detailed job info (very useful to diagnose issues)
`$ squeue --cluster=wice --long`	Status of all recent jobs
`$ squeue --cluster=wice --start`	Give a rough estimate of start time
`$ sinfo --cluster=wice`	Info about the state of available partitions and nodes
`$ sacct --cluster=wice --batch --job <JobID>`	Show minimal info about a queue or partition (-p)
`$ slurmtop` `$ scontrol --cluster=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

VLAAMS
SUPERCOMPUTER
CENTRUM

demo – conda installation

- ✅ Start an interactive job on Genius or wICE
    srun –M genius –A <account> -n 1 --pty /bin/bash -l
- ✅ Install miniconda in your $VSC_DATA directory
- $ wget https://repo.continuum.io/miniconda/Miniconda3-latest-Linux-x86_64.sh

$ bash Miniconda3-latest-Linux-x86_64.sh –b –p $VSC_DATA/miniconda3
- ✅ Add a PATH to conda:
- $ export PATH="${VSC_DATA}/miniconda3/bin:${PATH}"
- ✅ Check if conda is added to your $PATH ($ which conda)

    Add it to $PATH in your .bashrc
- $ echo 'export PATH="${VSC_DATA}/miniconda3/bin:${PATH}" ' >> .bashrc

VLAAMS
SUPERCOMPUTER
CENTRUM

demo – conda usage

- ✓ Create a conda environment including Jupyter

```
$ conda create -n science jupyter numpy scipy
```

- ✓ Activate this environment

```
$ source activate science
```

- ✓ Add matplotlib package to this environment

```
$ conda install matplotlib
```

- ✓ Return to original environment

```
$ conda deactivate
```

VLAAMS
SUPERCOMPUTER
CENTRUM

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 (User Day): https://www.vscentrum.be/vsctraining

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

VLAAMS
SUPERCOMPUTER
CENTRUM

Stay Connected to VSC

Start with VSC

LinkedIn

https://hpcleuven.github.io/HPC-intro/

SCAN ME

File   Edit   Selection   View   Go   Run   Terminal   Help

EXTENSIONS

Search Extensions in Marketplace

LOCAL - INSTALLED   5

Remote - SSH   160ms
Open any folder on a remote machi...
Microsoft

Remote - SSH: Editing Configurat...
Edit SSH configuration files
Microsoft

Remote - Tunnels
Connect to a remote machine throu...
Microsoft

Remote Explorer
View remote machines for SSH and ...
Microsoft

SSH: LOGIN.HPC.KULEUVEN.BE - INST...   0

RECOMMENDED   2

Debugger for Firefox   4.7M 4.5
Debug your web application or bro...
Firefox DevTools   Install

WSL   36.2M 5
Open any folder in the Windows Su...
Microsoft   Install

MCP SERVERS

MCP Servers

Untitled-1   testsheb.pbs   Exten

Select configured SSH host or enter user@host

login.hpc.kuleuven.be   Select configured SSH host or enter user@host
+ Add New SSH Host...
Configure SSH Hosts...

Microsoft microsoft.com | 29,189,049 | ★★★½☆ (203)

Open any folder on a remote machine using SSH and take advantage of VS Code's full feature set.

Uninstall   Switch to Pre-Release Version   ✓ Auto Update

DETAILS   FEATURES   EXTENSION PACK

Visual Studio Code Remote - SSH

The **Remote - SSH** extension lets you use any remote machine with a SSH server as your development environment. This can greatly simplify development and troubleshooting in a wide variety of situations. You can:

- Develop on the same operating system you deploy to or use larger, faster, or more specialized hardware than your local machine.
- Quickly swap between different, remote development environments and safely make updates without worrying about impacting your local machine.
- Access an existing development environment from multiple machines or locations.
- Debug an application running somewhere else such as a customer site or in the cloud.

No source code needs to be on your local machine to gain these benefits since the extension runs commands and other extensions directly on the remote machine. You can open any folder on the remote machine and work with it just as you would if the folder were on your own machine.

We're connected to our SSH host!

Marketplace

Identifier	ms-vscode-remote.remote-ssh
Version	0.120.0
Published	2019-05-02, 20:40:34
Last Released	2025-08-15, 17:21:42

Categories

Other

Resources

Marketplace
Issues
Repository
License
Microsoft

ⓘ Extension 'Remote - SSH' is required to open the remote window. ✕
Do you want to install the extension?

Install and Reload

⚠ Disconnected from SSH: login.hpc.kuleuven.be   ⊗ 0 ⚠ 0

/user/leuven/304/vsc30468/

OK Show Local

..
.ansys
.apptainer
.bash_histories
.cache
.conda
.config
.dbus
.emacs.d
.felix
.gnupg
.gvfs
.idl
.ipython
.irods

File   Edit   Selection   View   Go   ...

Welcome

Visual Stu
Editing evolved

Start

New File...

Open File...

Open Folder...

Clone Git Repository...

Connect to...

Generate New Workspace...

...ics, and start coding

Recent

monday   C:\Users\u0076109\OneDrive - KU Leuven\Downloads

Show welcome page on startup

SSH: login.hpc.kuleuven.be   0   0   0

ncview

- https://people.sc.fsu.edu/~jburkardt/data/netcdf/netcdf.html
- Xming+Enable X11 forwarding
- `$ module load ncview`

- `$wget https://people.sc.fsu.edu/~jburkardt/data/netcdf/simple_xy.nc`
- `$ ncview simple_xy.nc`