

# VSC HPC Introduction

ICTS KU Leuven

VSC staff:

Mag Selwa

Ehsan Moravveji

Wouter van Assche

Geert Jan Bex (UHasselt)

Jan Ooghe



# 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

# VSC PARTNERSHIP





Surf to: [www.vscentrum.be](http://www.vscentrum.be)

Very rich suite of  
courses every  
academic year

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





## Welcome to the User Portal

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

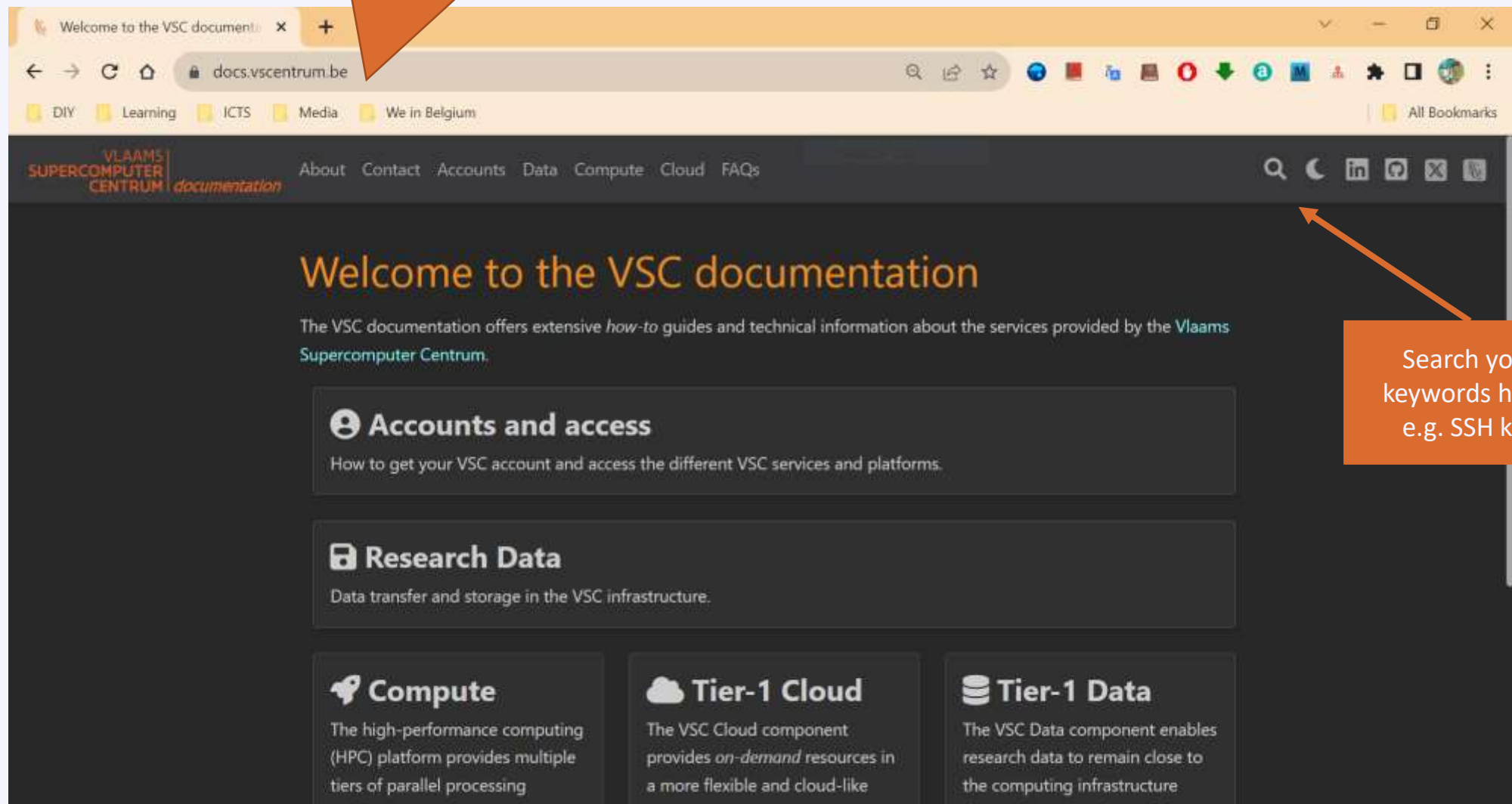
[User documentation page](#)

**Manage your VSC-account**

*partner institute account required*

[VSC account page](#)

<https://docs.vscentrum.be>



The screenshot shows a web browser window with the URL [docs.vscentrum.be](https://docs.vscentrum.be). The browser's address bar and tabs are visible at the top. Below the browser window, the website's header features the VLAAMS SUPERCOMPUTER CENTRUM logo and navigation links: About, Contact, Accounts, Data, Compute, Cloud, and FAQs. A search bar is located in the top right corner of the website header, with an orange arrow pointing to it from a callout box. The main content area has a dark background and features a large heading "Welcome to the VSC documentation" followed by a paragraph: "The VSC documentation offers extensive *how-to* guides and technical information about the services provided by the Vlaams Supercomputer Centrum." Below this, there are three main sections: "Accounts and access" (with a user icon), "Research Data" (with a document icon), and a row of three sections: "Compute" (with a rocket icon), "Tier-1 Cloud" (with a cloud icon), and "Tier-1 Data" (with a database icon). Each section includes a brief description of the service.

**VLAAMS SUPERCOMPUTER CENTRUM** documentation

About Contact Accounts Data Compute Cloud FAQs

# Welcome to the VSC documentation

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

## Accounts and access

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

## Research Data

Data transfer and storage in the VSC infrastructure.

### Compute

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

### Tier-1 Cloud

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

### Tier-1 Data

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

Search your keywords here;  
e.g. SSH key

To manage your  
VSC account:

[account.vscentrum.be](https://account.vscentrum.be)

The screenshot shows a web browser window with the URL [account.vscentrum.be/django/](https://account.vscentrum.be/django/) in the address bar. The page features the VLAAMS SUPERCOMPUTER CENTRUM and Vlaanderen is supercomputing logos. A navigation bar contains buttons: View Account, Edit Account, View Groups, New/Join Group, Edit Group, New/Join VO, Reservations, and Log Out. The 'View Account' button is highlighted with an orange callout. Below the navigation bar, the 'View account' section is visible, with 'General information' expanded. This section displays fields for Uid, Institute (Leuven / Hasselt), Institute login, and Gecos. Three orange callouts are present: one pointing to the 'Edit Account' button with the text 'Upload SSH Key Request account'; another pointing to the 'New/Join Group' button with the text 'Create or request to join a new group'; and a third pointing to the 'Edit Group' button with the text 'Moderate your own groups (add/remove users)'. A fourth orange callout points to the address bar with the text 'Upload more SSH public keys'.

View account — VSC

account.vscentrum.be/django/

VLAAMS SUPERCOMPUTER CENTRUM Vlaanderen is supercomputing

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

View account

General information

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

Upload SSH Key Request account

Upload more SSH public keys

Create or request to join a new group

Moderate your own groups (add/remove users)



# Support and Services

## Basic support

- Helpdesk ([hpcinfo@kuleuven.be](mailto:hpcinfo@kuleuven.be))
- Monitoring and reporting

## Application support

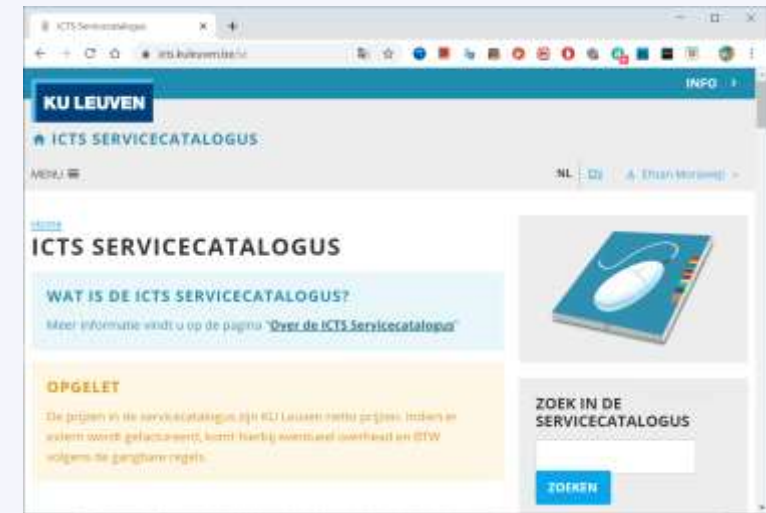
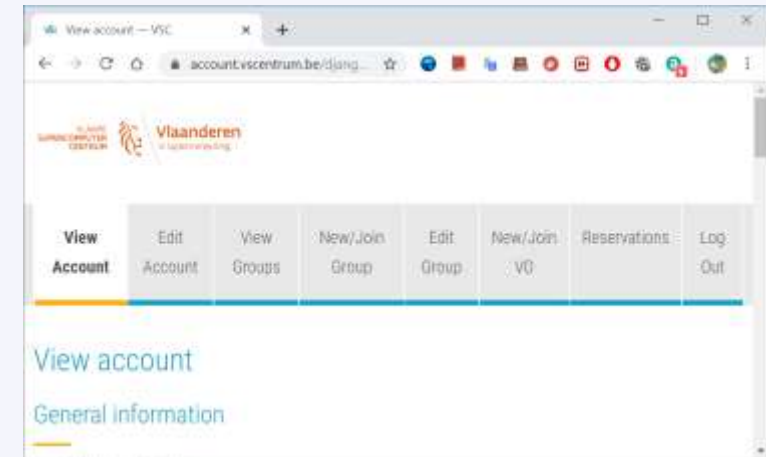
- Installation and porting
- Optimisation and debugging
- Benchmarking
- Workflows and best practices

## Training

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

# Become a VSC user

- ❑ (Optional) Create a secure (4096 bit) [SSH key pairs](#)  
Upload it on the account page: [www.account.vscentrum.be](http://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)



# VSC training

- Introductory

Matlab (Supercalculator)  
Matlab Programming

Linux

HPC intro

Linux for HPC

Make intro

CMake intro

Linux scripting

Linux tools

Version control systems

Infosessions:

- Containers
- Notebooks

- Intermediate

C++ for scientific computing

Fortran for programmers

C

- 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>

# To Acknowledge VSC in publications

## Why?

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

## At KU Leuven

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

## In het nederlands

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

## In English

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



## Tier-2 Clusters



# VSC HPC Environments



# Tier-2 Clusters @ KU Leuven

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



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



# Tier-2 Cluster - Genius

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

- **End of life:** end 2024
- **Access:** login.hpc.kuleuven.be
- **Interconnect:** InfiniBand EDR (25 Gb/s)
- **SSD Disks:** 200 GB

Remarks

# 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 <b>large mem</b>	Xeon 8360Y	72	2 TB	28000	bigmem
Icelake <b>large mem</b>	Xeon 8360Y	72	8 TB	111900	hugemem
Icelake <b>GPU</b>	Xeon 8360Y 4xA100 SXM2 80GB	72	512 GB	7000	gpu_a100
AMD <b>GPU</b>	Epyc 9334 4xH100 SXM5 80GB	64	768 GB	11700	gpu_h100
Icelake <b>Interactive</b>	Xeon 8358 1xA100 SXM2 80GB	64	512 GB	7500	interactive

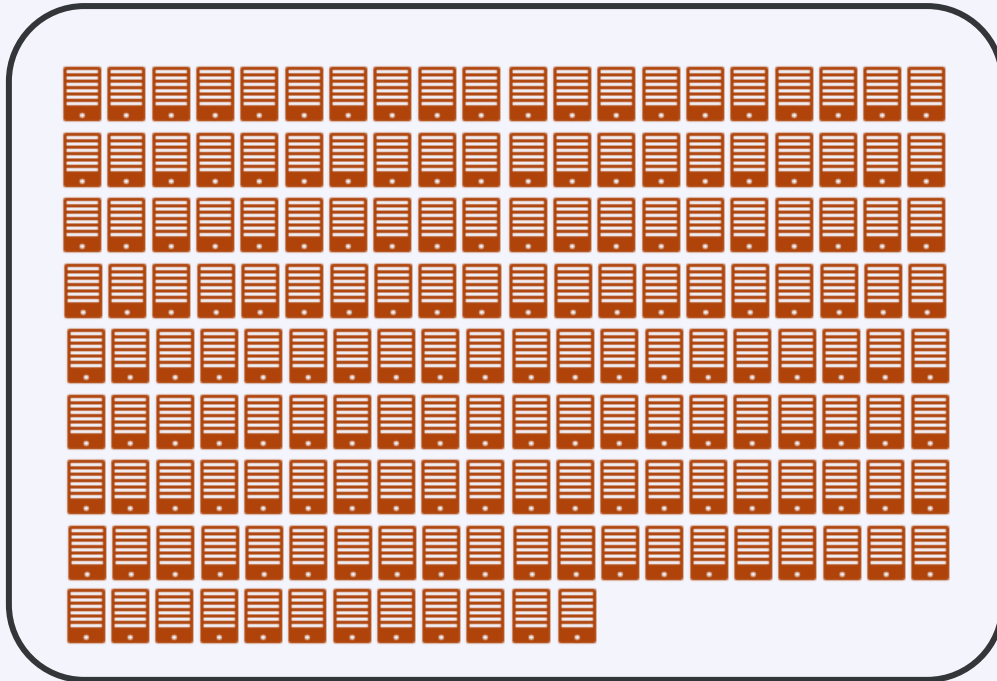
- **Phase 1:** since September 2022
- **Access:** login.hpc.kuleuven.be
- **Interconnect:** InfiniBand HDR (100 Gb/s)

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

Remarks

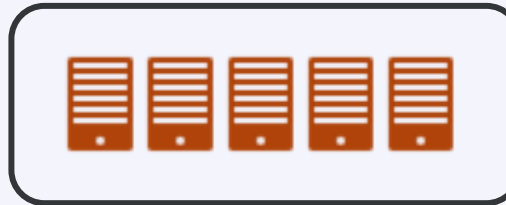
# Tier-2 Cluster - wICE

## Compute nodes



172x IceLake 72c 256 GB

## Large memory nodes



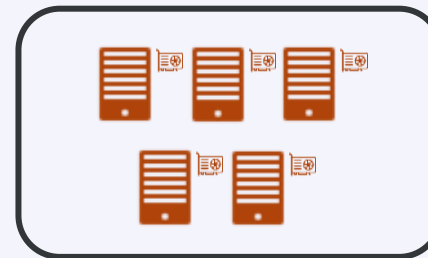
5x IceLake 72c 2048 GB

## GPU nodes



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

## Interactive nodes



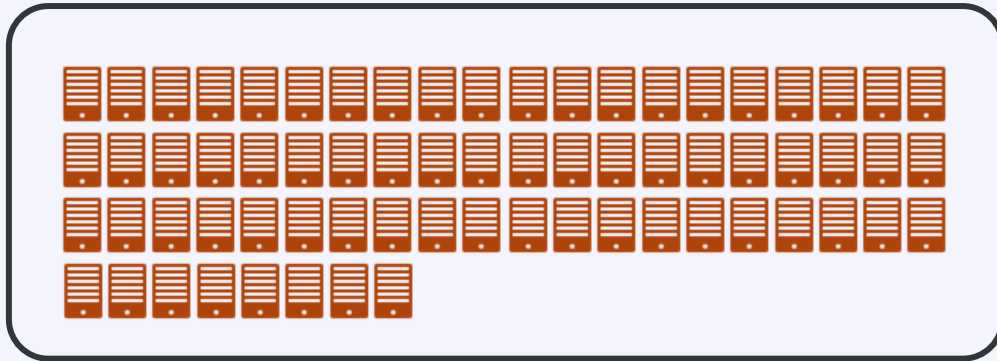
4x IceLake 64c 512 GB  
1 A100 80GB

**No Dedicated  
Login Nodes**



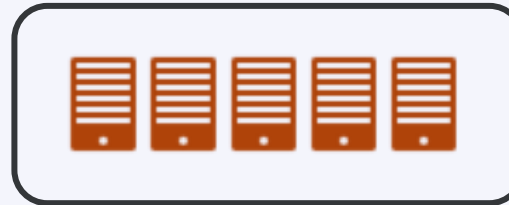
# Tier-2 Cluster – wICE extension

## Compute nodes



68x Sapphire Rapids 96c 256 GB

## Large memory nodes



1x IceLake 72c 8 TB

## GPU nodes



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

The background is a vibrant, stylized illustration. On the left, a cityscape features large, grey, arched structures. A digital screen in the center shows a grid with numbers and a large orange arrow pointing upwards. On the right, a close-up of a plant with large blue and red berries is visible. A large, semi-transparent orange rectangle covers the lower-left portion of the image, containing the word 'Storage' in white text.

# Storage

# Overview of the storage infrastructure

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

# Storage

- ❑ **Do NOT use /tmp**

It is only 10 GB and is reserved for the OS and root processes.

Your application can crash if using /tmp

- ❑ You are automatically logged into your home folder upon login.

Immediately go to your other storages, e.g.

```
$ cd $VSC_DATA
```

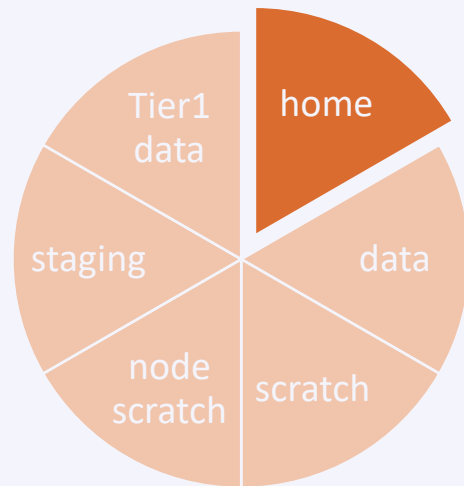
- ❑ Always check your storage balance using `myquota` command

## Example

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

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

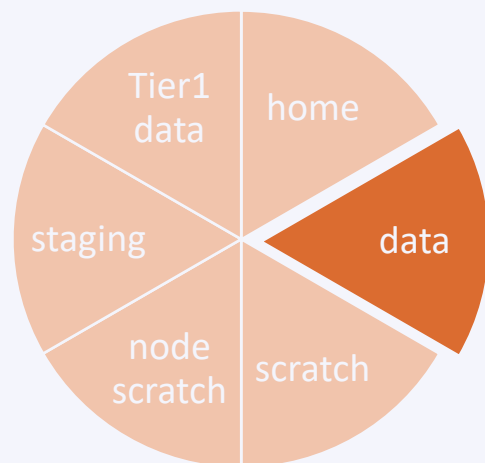
# Storage



Storage	home folder
Env. Variable	<code>\$VSC_HOME</code>
Filesystem Type	NFS
Access	Global
Backup	Hourly, daily, weekly (until last month) inside the <code>.snapshot</code> folder.
Default Quota	3 GB
Extension	Not possible
Usage	Only storing SSH keys, config files
Remarks	<ul style="list-style-type: none"><li>- Stay away from using it</li><li>- Can easily overflow:<ul style="list-style-type: none"><li>+ Your jobs may crash</li><li>+ Login issues</li></ul></li></ul>

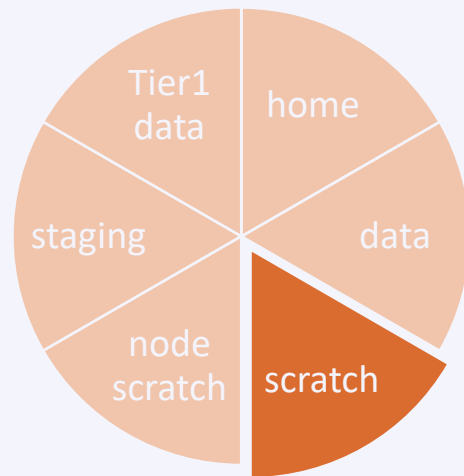


# Storage



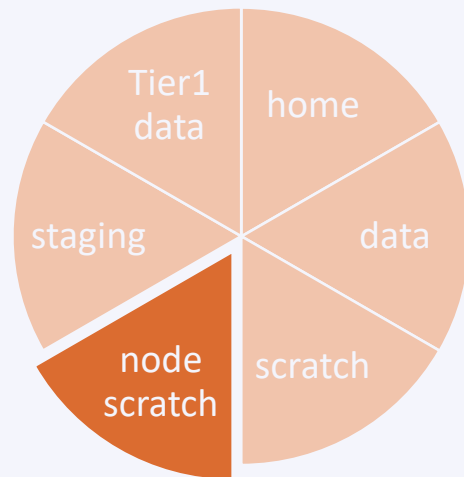
Storage	data folder
Env. Variable	\$VSC_DATA
Filesystem Type	NFS
Access	Global
Backup	Hourly, daily, weekly (until last month) inside the .snapshot folder.
Default Quota	75 GB
Extension	On purchase
Usage	Your data, code, software, results
Remarks	<ul style="list-style-type: none"> <li>- Permanent storage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

# Storage



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

# Storage

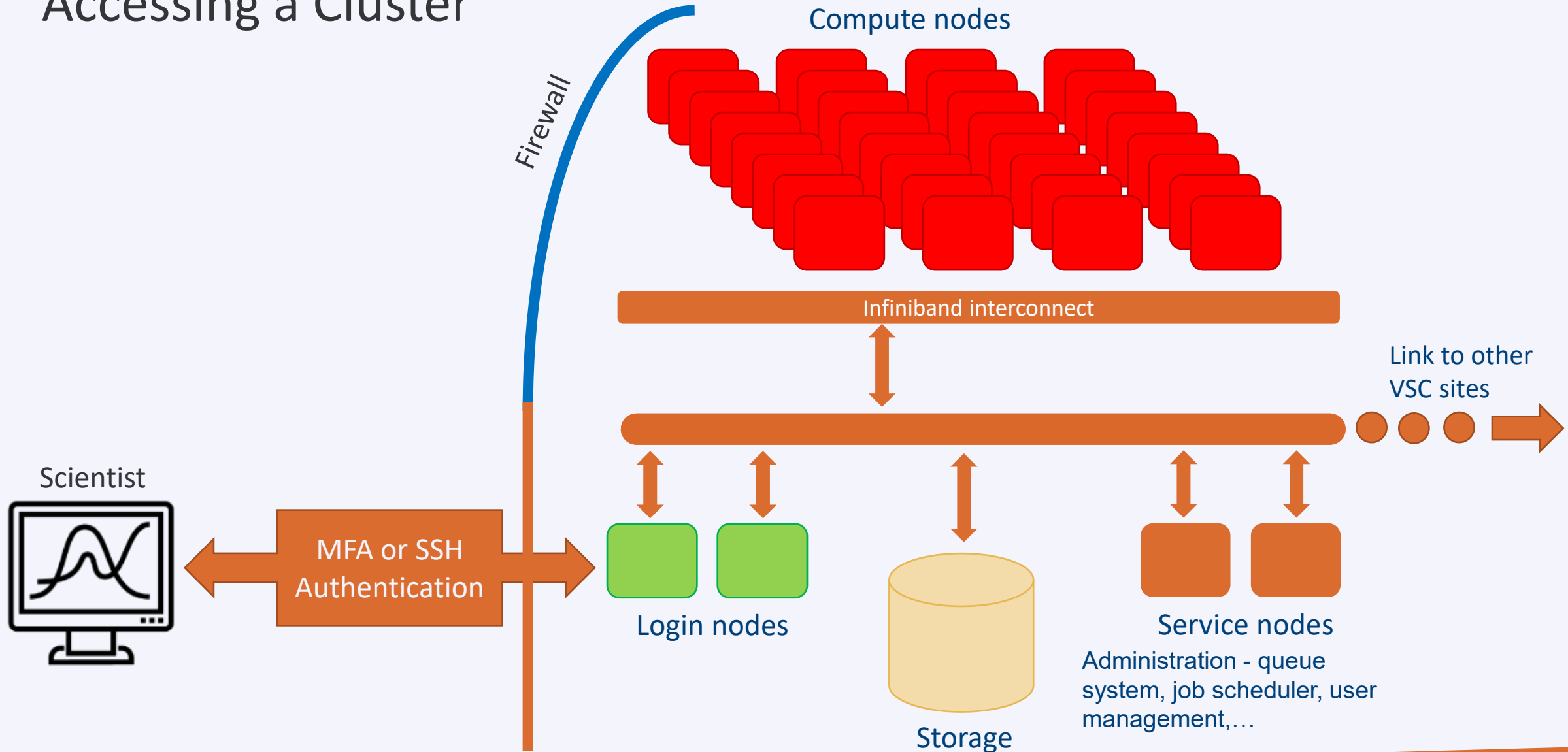


Storage	Node scratch folder
Env. Variable	<code>\$VSC_SCRATCH_NODE</code>
Filesystem Type	Lustre
Access	On compute node, only at runtime
Backup	None
Default Quota	591 GB
Extension	<a href="#">Read about beeOND</a>
Usage	Temporary storage at runtime
Remarks	<ul style="list-style-type: none"><li>- Fastest I/O, attached to the node</li><li>- Is cleaned after job terminates</li><li>- Copy the data to your home, scratch, or staging before job ends</li></ul>



## Login Nodes & MFA

# Accessing a Cluster





# How to login?

## Open OnDemand

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

## NX

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

## SSH client

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

# Using Login Nodes

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

## Tips

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

## Warning

Login nodes are shared resources

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

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

# 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

# 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
```

# Generate the key

Linux/Mac OS users:

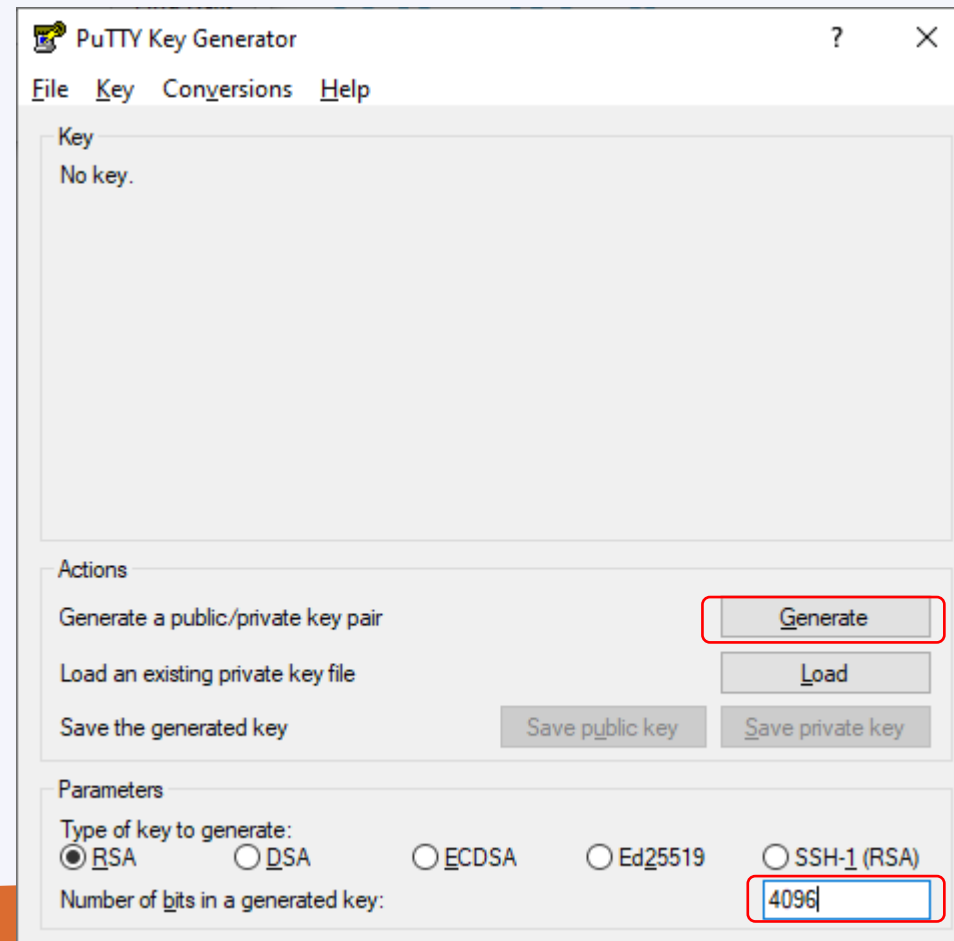
- Use “*ssh-keygen* ” *command to generate key pair*
- **Be sure to give your key a passphrase!**
- Default location: `~/.ssh/id_rsa (~/.ssh/id_rsa.pub)`
- If other location: `ssh -i location-of-the-file .....`
- **keychain:** ssh agent to load the key with passphrase for current linux session



# Generate the key

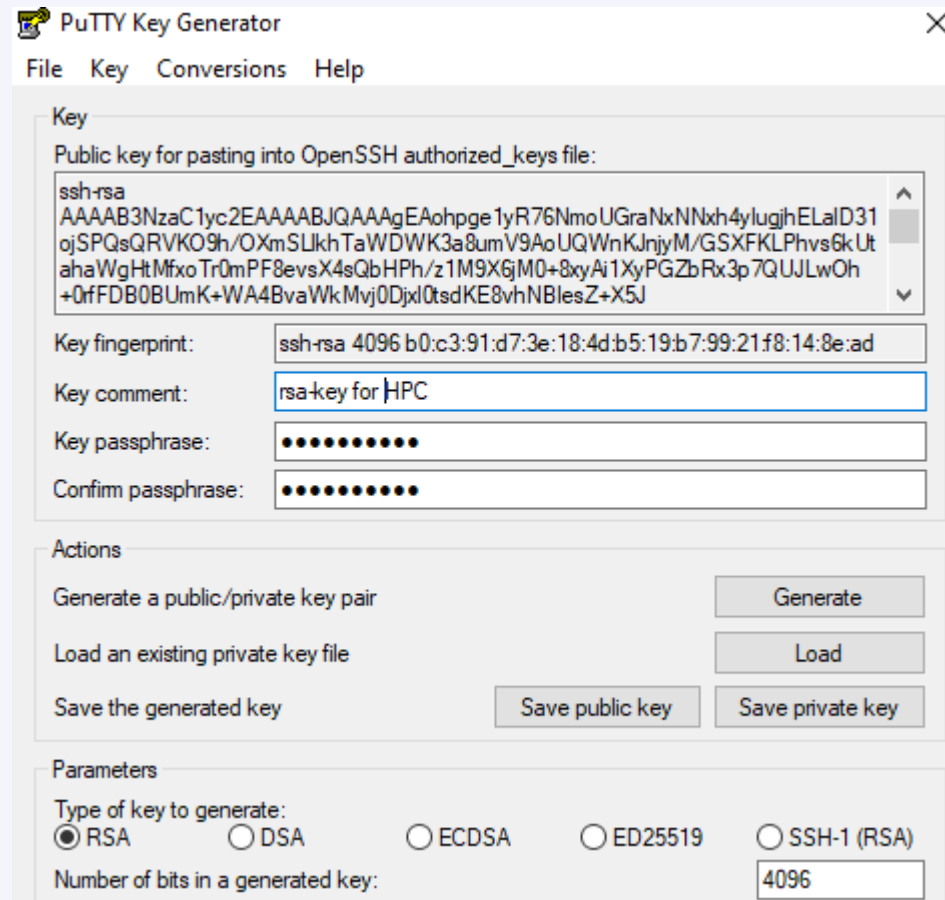
Windows users: use the PuTTYgen key generator.

Request ssh key format: RSA 4096 bit



# Generate the key

Be sure to give your key a passphrase!



The screenshot shows the PuTTY Key Generator window. The 'Key' section displays the public key for pasting into the OpenSSH authorized\_keys file. The key fingerprint is shown as 'ssh-rsa 4096 b0:c3:91:d7:3e:18:4d:b5:19:b7:99:21:f8:14:8e:ad'. The key comment is 'rsa-key for HPC'. The key passphrase is masked with dots. The 'Actions' section includes buttons for 'Generate', 'Load', 'Save public key', and 'Save private key'. The 'Parameters' section shows the 'Type of key to generate' set to 'RSA' and the 'Number of bits in a generated key' set to '4096'.

PuTTY Key Generator

File Key Conversions Help

Key

Public key for pasting into OpenSSH authorized\_keys file:

```
ssh-rsa
AAAAB3NzaC1yc2EAAAABJQAAAgEAohpge1yR76NmoUGraNxNNxh4ylugjhELaID31
ojSPQsQRVKO9h/OXmSLkhTaWDWK3a8umV9AoUQWnKJnlyM/GSXFkLPhvs6kUt
ahaWgHtMfxoTr0mPF8evsX4sQbHPh/z1M9XGjM0+8xyAi1XyPGZbRx3p7QUJLwOh
+QfFDB0BUmK+WA4BvaWkMvj0Djxl0tsdKE8vhNBlesZ+X5J
```

Key fingerprint: ssh-rsa 4096 b0:c3:91:d7:3e:18:4d:b5:19:b7:99:21:f8:14:8e:ad

Key comment: rsa-key for HPC

Key passphrase: .....

Confirm passphrase: .....

Actions

Generate a public/private key pair Generate

Load an existing private key file Load

Save the generated key Save public key Save private key

Parameters

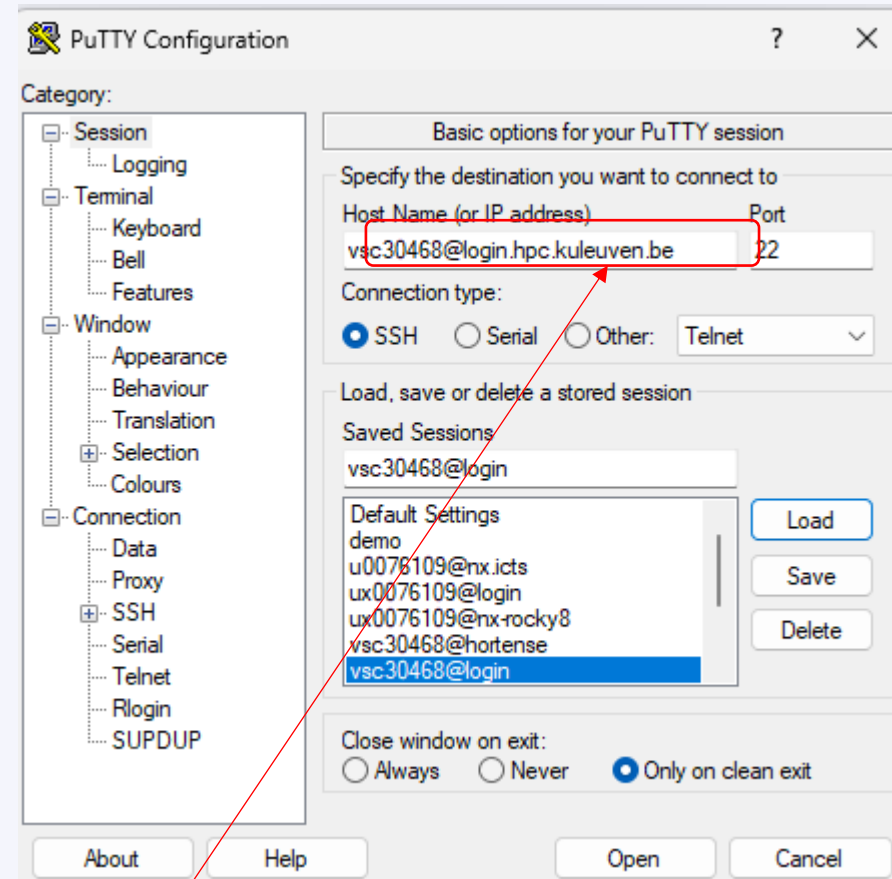
Type of key to generate:  
☒ RSA ☐ DSA ☐ ECDSA ☐ ED25519 ☐ SSH-1 (RSA)

Number of bits in a generated key: 4096

# 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.

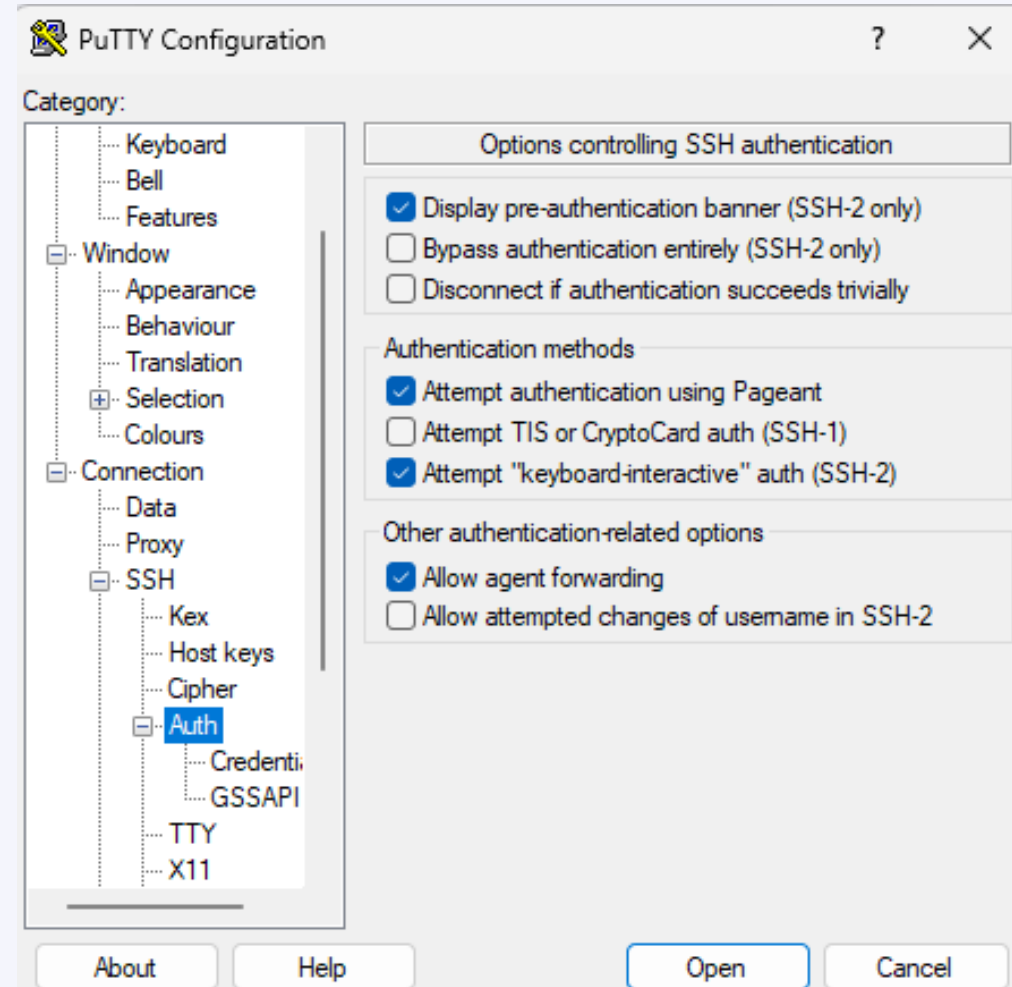


vsc3XXXX@login.hpc.kuleuven.be

# Connecting to the cluster: text mode

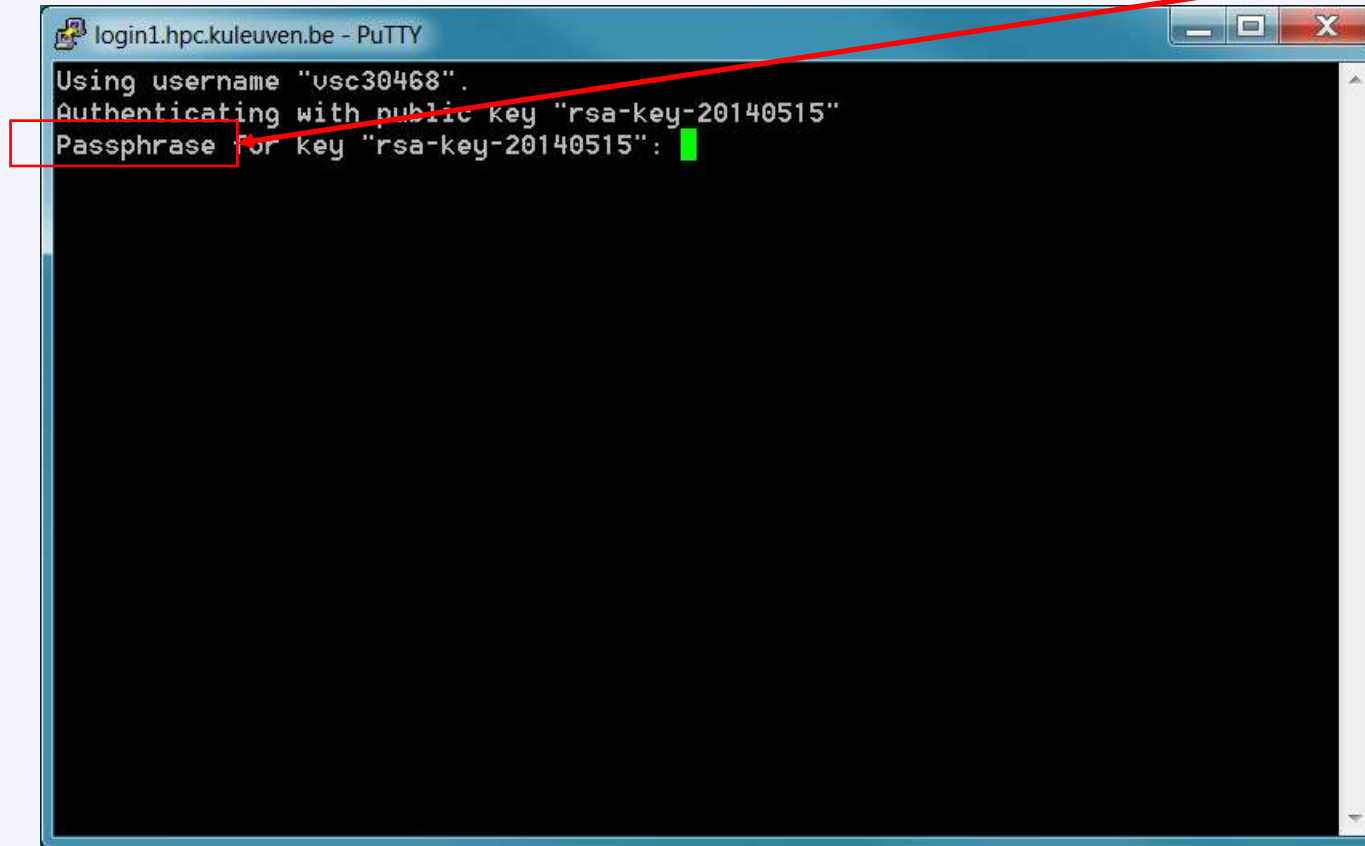
## Windows users:

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



# Connecting to the cluster: text mode

Windows users:



```
login1.hpc.kuleuven.be - PuTTY
Using username "usc30468".
Authenticating with public key "rsa-key-20140515"
Passphrase for key "rsa-key-20140515": █
```

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 vscXXXXXX@<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[redacted] — ssh vsc[redacted]@login.hpc.kuleuven.be — 80x25
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:~ [redacted]$ ssh vsc[redacted]@login.hpc.kuleuven.be
[Enter passphrase for key '/Users/[redacted]/.ssh/id_ed25519':
vsc[redacted]
Please open https://firewall.vscenrum.be/PDihC0i5kk9Sks3VN5m to complete logon.
Last login: Fri Oct 7 11:32:48 2022 from 2a02:2c40:0:2410::a1b

Genius

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[redacted]@tier2-p-login-4 ~ $
```

# Connecting via Terminal

(Linux and Mac)

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

## With SSH Agent

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



```
u[REDACTED] — ssh vsc[REDACTED]@login.hpc.kuleuven.be — 80x25
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:~ [REDACTED]$ ssh vsc[REDACTED]@login.hpc.kuleuven.be
[Enter passphrase for key '/Users/[REDACTED]/.ssh/id_ed25519':
vsc[REDACTED]
Please open https://firewall.vscenrum.be/PDihC0i5kk9Sks3VN5m to complete logon.
Last login: Fri Oct  7 11:32:48 2022 from 2a02:2c40:0:2410::a1b

  ( )
Genius

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[REDACTED]@tier2-p-login-4 ~ $
```

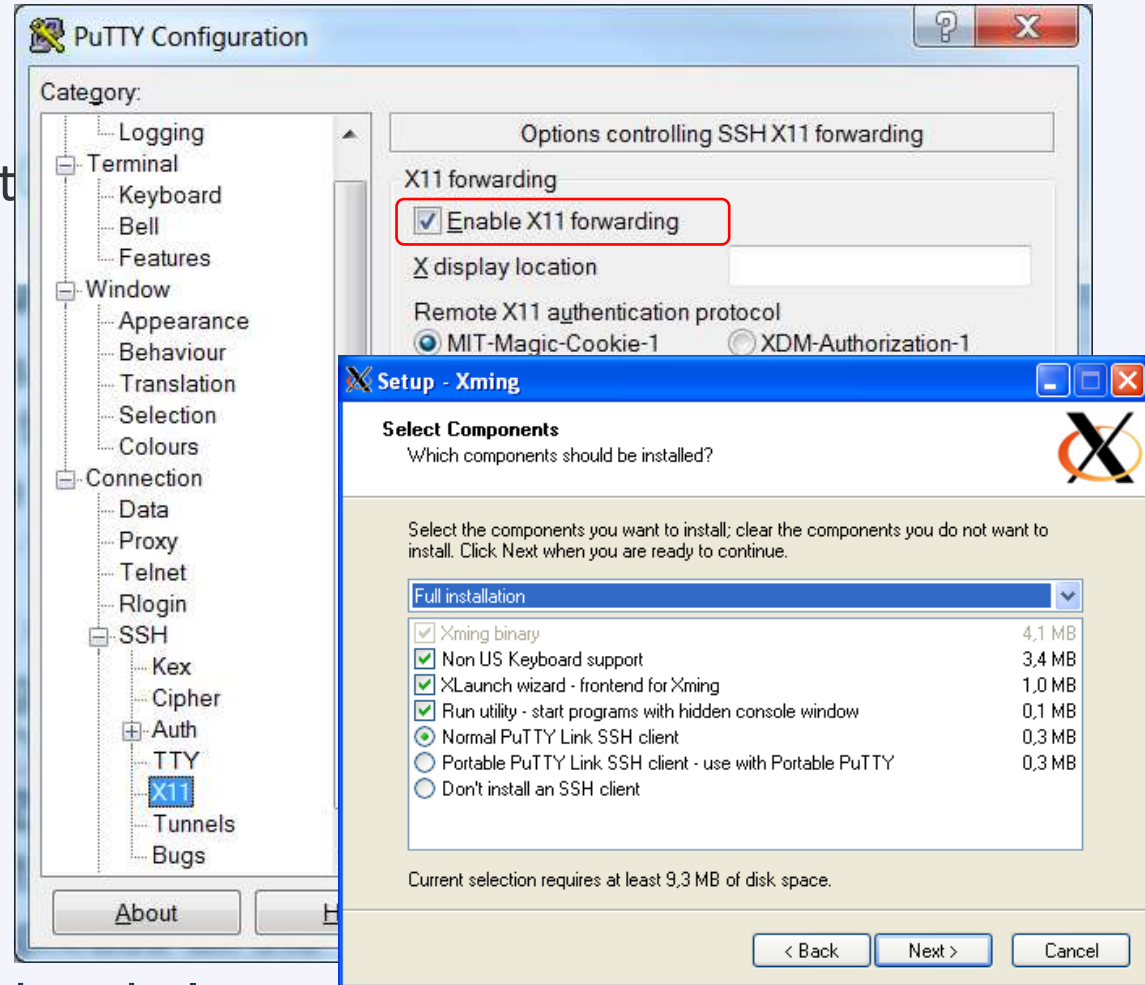
# Connecting to the cluster: display graphics

## 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

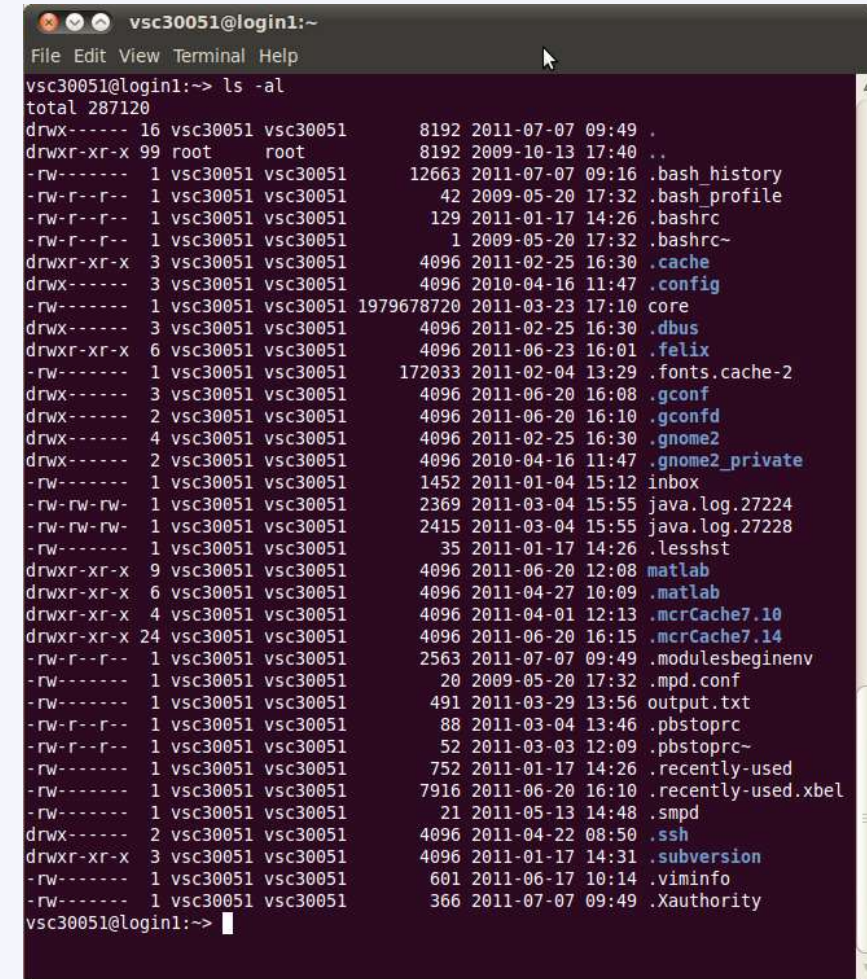
## Linux users:

- `ssh -X vsc3XXXX@login.hpc.kuleuven.be`



# 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**



```
vsc30051@login1:~  
File Edit View Terminal Help  
vsc30051@login1:~> ls -al  
total 287120  
drwx----- 16 vsc30051 vsc30051      8192 2011-07-07 09:49 .  
drwxr-xr-x 99 root      root          8192 2009-10-13 17:40 ..  
-rw----- 1 vsc30051 vsc30051    12663 2011-07-07 09:16 .bash_history  
-rw-r--r-- 1 vsc30051 vsc30051      42 2009-05-20 17:32 .bash_profile  
-rw-r--r-- 1 vsc30051 vsc30051     129 2011-01-17 14:26 .bashrc  
-rw-r--r-- 1 vsc30051 vsc30051      1 2009-05-20 17:32 .bashrc~  
drwxr-xr-x 3 vsc30051 vsc30051     4096 2011-02-25 16:30 .cache  
drwx----- 3 vsc30051 vsc30051     4096 2010-04-16 11:47 .config  
-rw----- 1 vsc30051 vsc30051 1979678720 2011-03-23 17:10 core  
drwx----- 3 vsc30051 vsc30051     4096 2011-02-25 16:30 .dbus  
drwxr-xr-x 6 vsc30051 vsc30051     4096 2011-06-23 16:01 .felix  
-rw----- 1 vsc30051 vsc30051    172033 2011-02-04 13:29 .fonts.cache-2  
drwx----- 3 vsc30051 vsc30051     4096 2011-06-20 16:08 .gconf  
drwx----- 2 vsc30051 vsc30051     4096 2011-06-20 16:10 .gconfd  
drwx----- 4 vsc30051 vsc30051     4096 2011-02-25 16:30 .gnome2  
drwx----- 2 vsc30051 vsc30051     4096 2010-04-16 11:47 .gnome2_private  
-rw----- 1 vsc30051 vsc30051    1452 2011-01-04 15:12 inbox  
-rw-rw-rw- 1 vsc30051 vsc30051    2369 2011-03-04 15:55 java.log.27224  
-rw-rw-rw- 1 vsc30051 vsc30051    2415 2011-03-04 15:55 java.log.27228  
-rw----- 1 vsc30051 vsc30051      35 2011-01-17 14:26 .lessht  
drwxr-xr-x 9 vsc30051 vsc30051     4096 2011-06-20 12:08 matlab  
drwxr-xr-x 6 vsc30051 vsc30051     4096 2011-04-27 10:09 .matlab  
drwxr-xr-x 4 vsc30051 vsc30051     4096 2011-04-01 12:13 .mcrCache7.10  
drwxr-xr-x 24 vsc30051 vsc30051     4096 2011-06-20 16:15 .mcrCache7.14  
-rw-r--r-- 1 vsc30051 vsc30051    2563 2011-07-07 09:49 .modulesbeginenv  
-rw----- 1 vsc30051 vsc30051      20 2009-05-20 17:32 .mpd.conf  
-rw----- 1 vsc30051 vsc30051     491 2011-03-29 13:56 output.txt  
-rw-r--r-- 1 vsc30051 vsc30051      88 2011-03-04 13:46 .pbstoprc  
-rw-r--r-- 1 vsc30051 vsc30051      52 2011-03-03 12:09 .pbstoprc~  
-rw----- 1 vsc30051 vsc30051     752 2011-01-17 14:26 .recently-used  
-rw----- 1 vsc30051 vsc30051    7916 2011-06-20 16:10 .recently-used.xbel  
-rw----- 1 vsc30051 vsc30051      21 2011-05-13 14:48 .smpd  
drwx----- 2 vsc30051 vsc30051     4096 2011-04-22 08:50 .ssh  
drwxr-xr-x 3 vsc30051 vsc30051     4096 2011-01-17 14:31 .subversion  
-rw----- 1 vsc30051 vsc30051      601 2011-06-17 10:14 .viminfo  
-rw----- 1 vsc30051 vsc30051      366 2011-07-07 09:49 .Xauthority  
vsc30051@login1:~> |
```

# 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

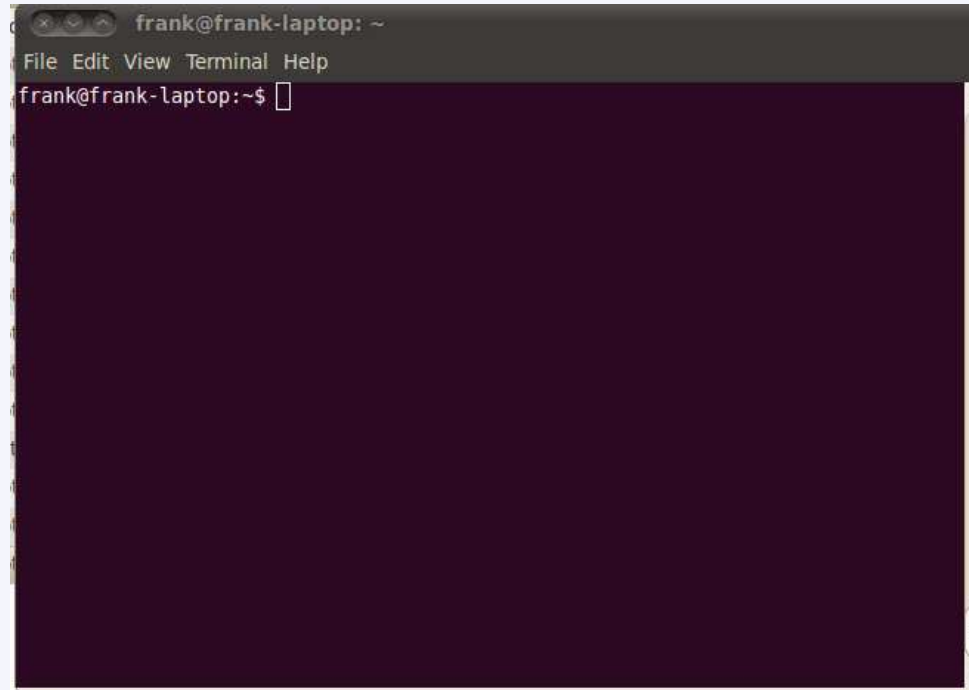


# Important rules on the command line

- “\” 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 themselves
- 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

# 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
```



# 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.

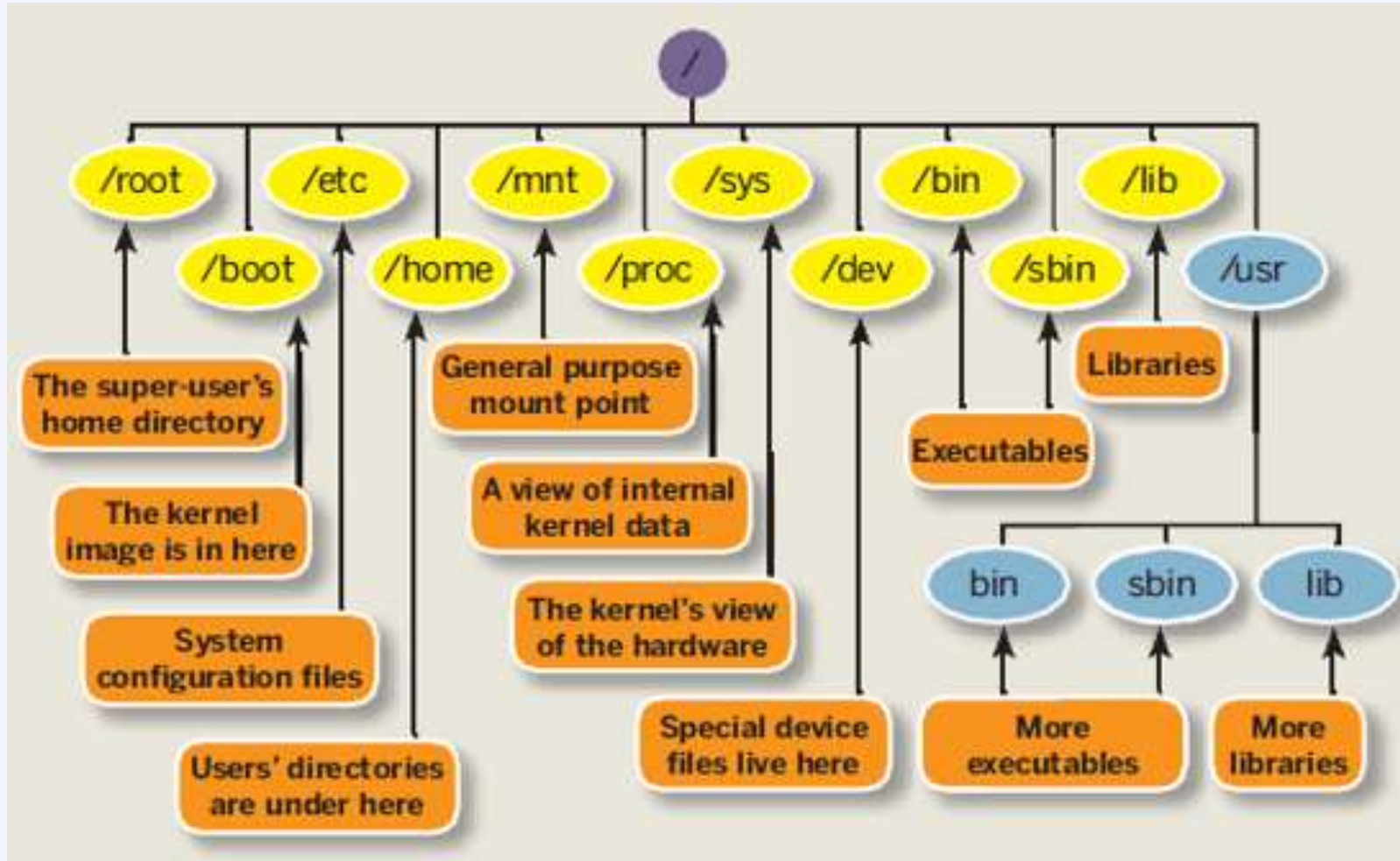
- `$ exit`

- 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.

# 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 /*: the 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

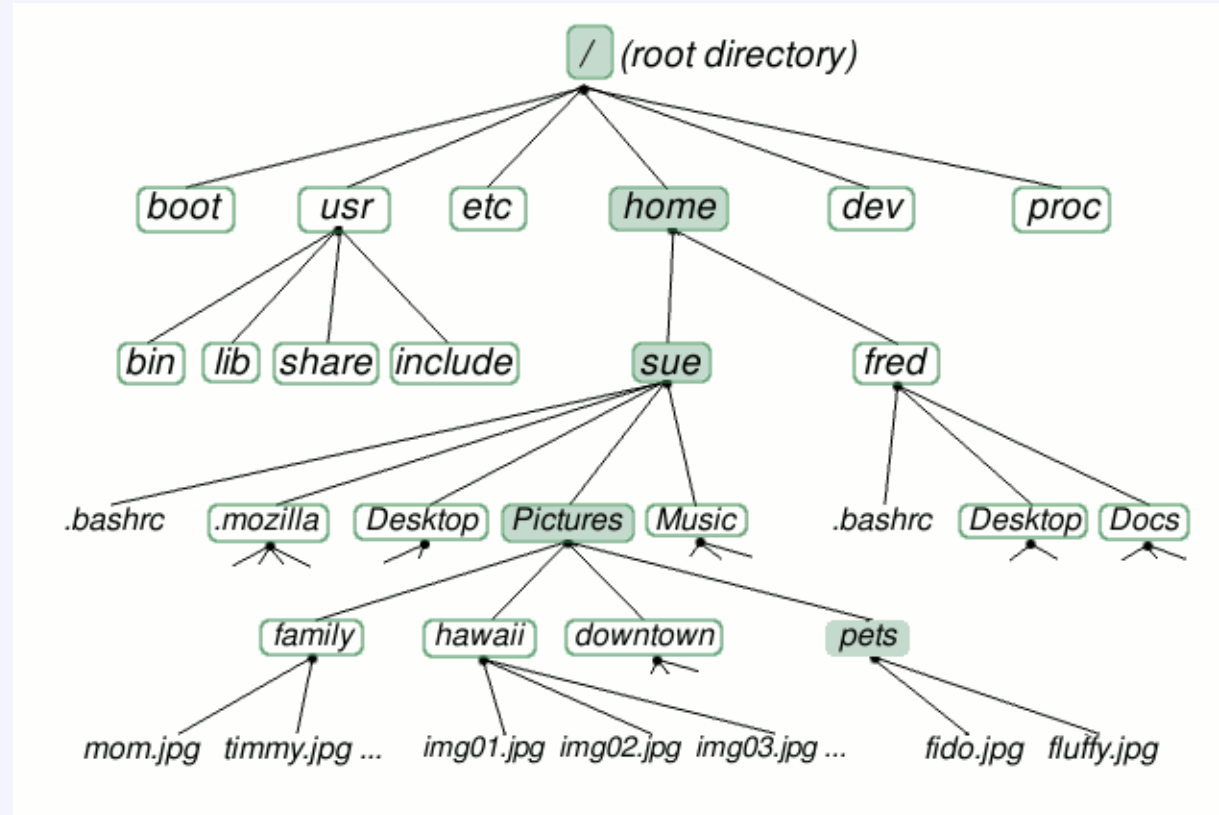
<code>/lost+found/</code>	Corrupt files the system tried to recover
<code>/mnt/</code>	Mount points for temporarily mounted filesystems
<code>/opt/</code>	Specific tools installed by the sysadmin
<code>/usr/local/</code>	often used instead
<code>/proc/</code>	Access to system information
<code>/sbin/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

# 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

# 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

The terminal window shows the user 'dlun' at 'enpklun.polyu.edu.hk' in the directory '/home/dlun/Desktop'. The command `ln -s ../CUI anotherCUI` is executed and circled in red. Below it, the command `ls -al` is executed, showing a detailed listing of files and directories. The output includes permissions, owner, group, size, date, and file name. Annotations with red arrows point to specific parts of the output: one points to the permissions of the symbolic link 'anotherCUI' (labeled 'A symbolic link begins with a letter /'), and another points to the blue-colored directory names (labeled 'Names in blue are directories, indicated by a letter d at the beginning of the line').

```
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.kdeInk
-rw-r--r--  1 dlun  dlun   159 May 17 2001 Red Hat Errata.kdeInk
-rw-r--r--  1 dlun  dlun   153 May 17 2001 Red Hat Support.kdeInk
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.kdeInk
-rw-r--r--  1 dlun  dlun   395 May 17 2001 floppy.kdeInk
-rw-r--r--  1 dlun  dlun   144 May 17 2001 www.redhat.com.kdeInk
[dlun@enpklun Desktop]$
```

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

A symbolic link begins with a letter /

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

# Moving around

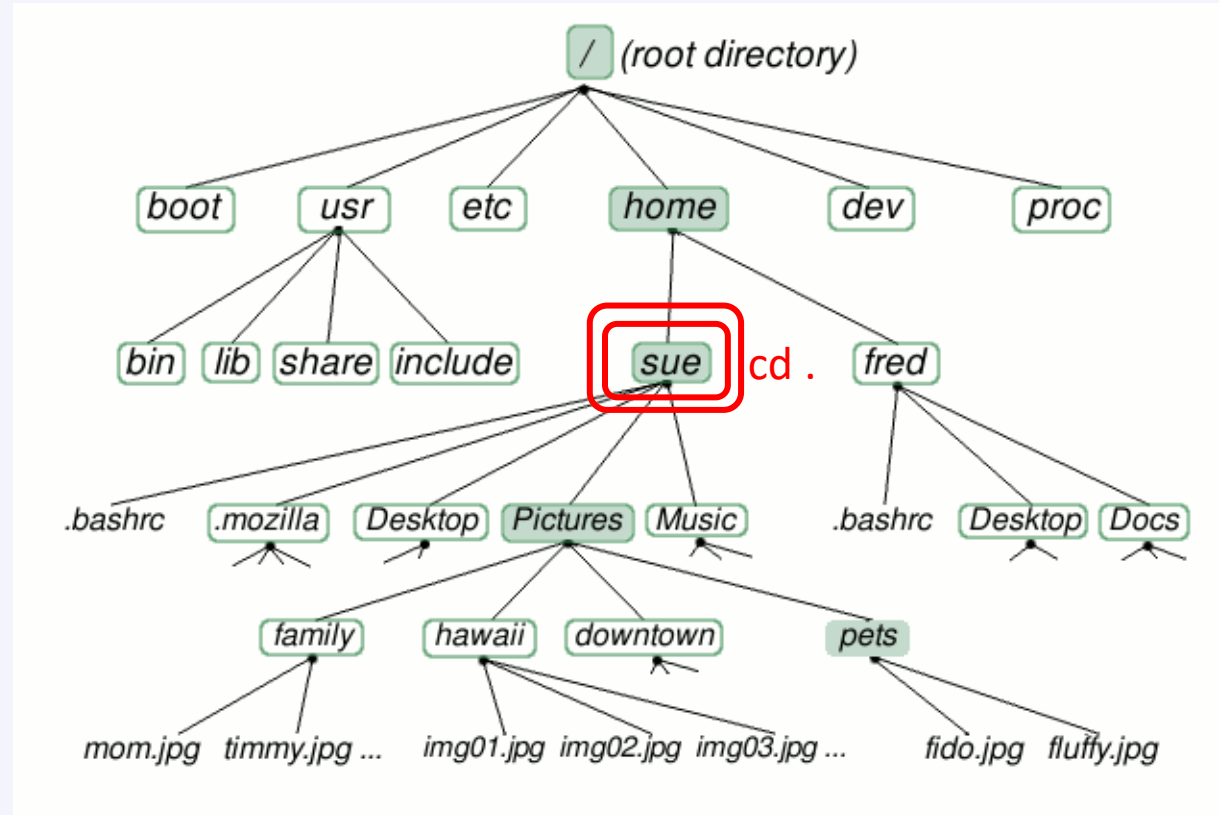
- Display the current/working directory
  - `$ pwd`
  - **Print Working Directory**
  - displays your current location within the file system.
- Change (navigate) directories.
  - `$ cd dir_name`
  - **Change Directories**
  - changes the position to the specific directory
- You can specify directory 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`

# 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

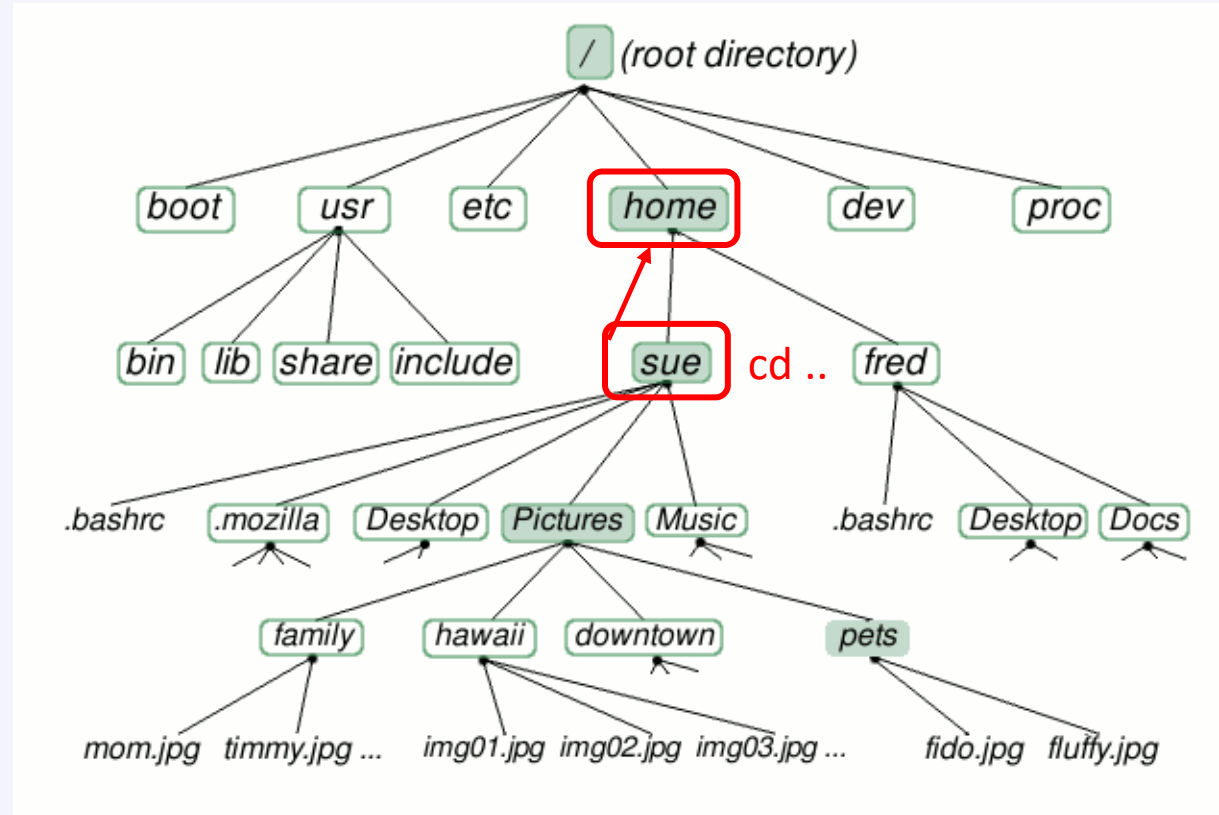


# Linux File System - directories



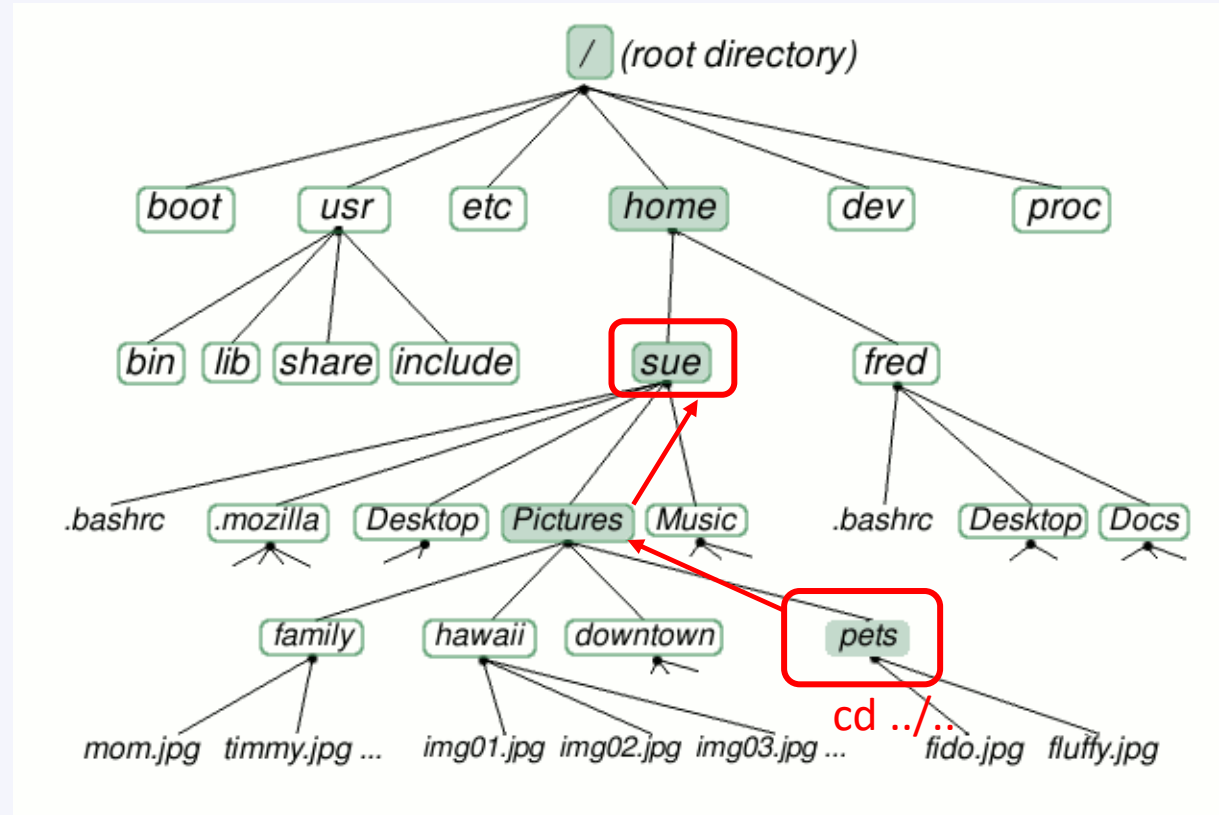
Source: <http://www.linuxplanet.com/linuxplanet/tutorials/6666/1/screenshot3894/>

# Linux File System - directories



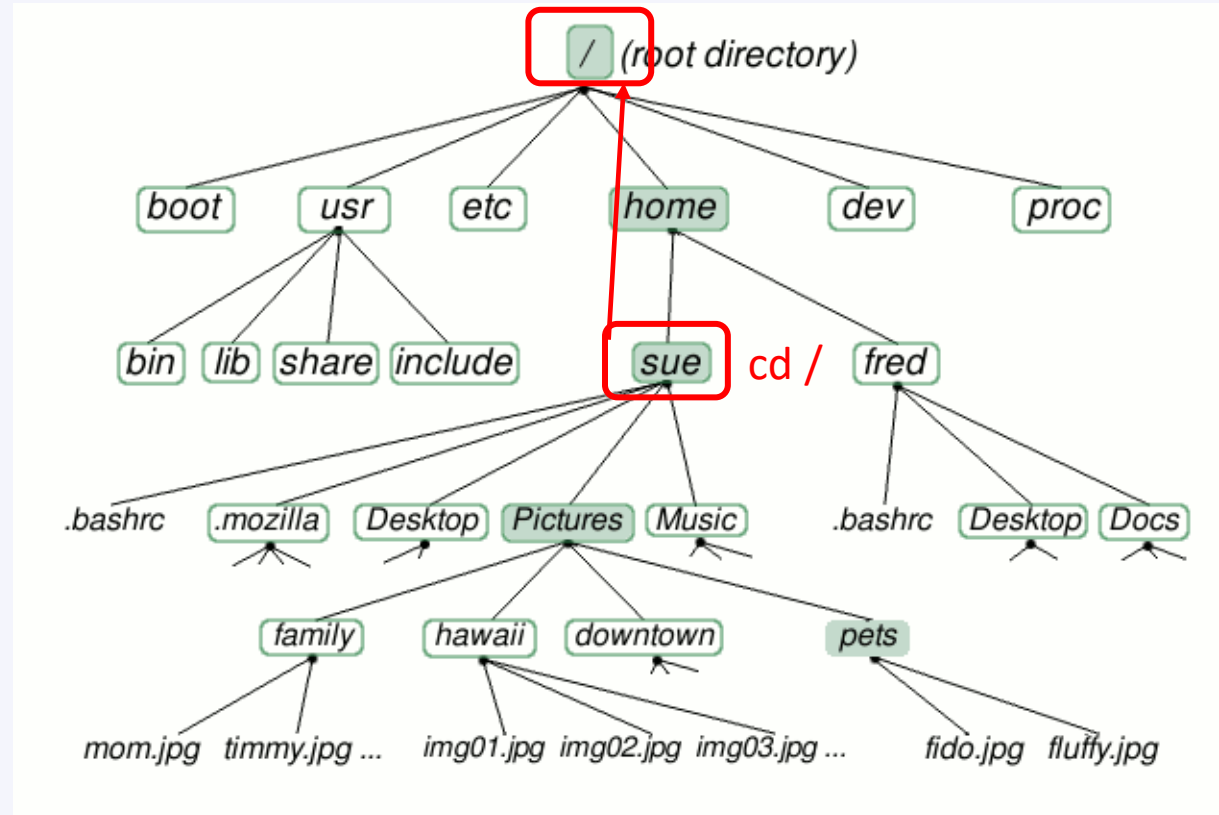
Source: <http://www.linuxplanet.com/linuxplanet/tutorials/6666/1/screenshot3894/>

# Linux File System - directories



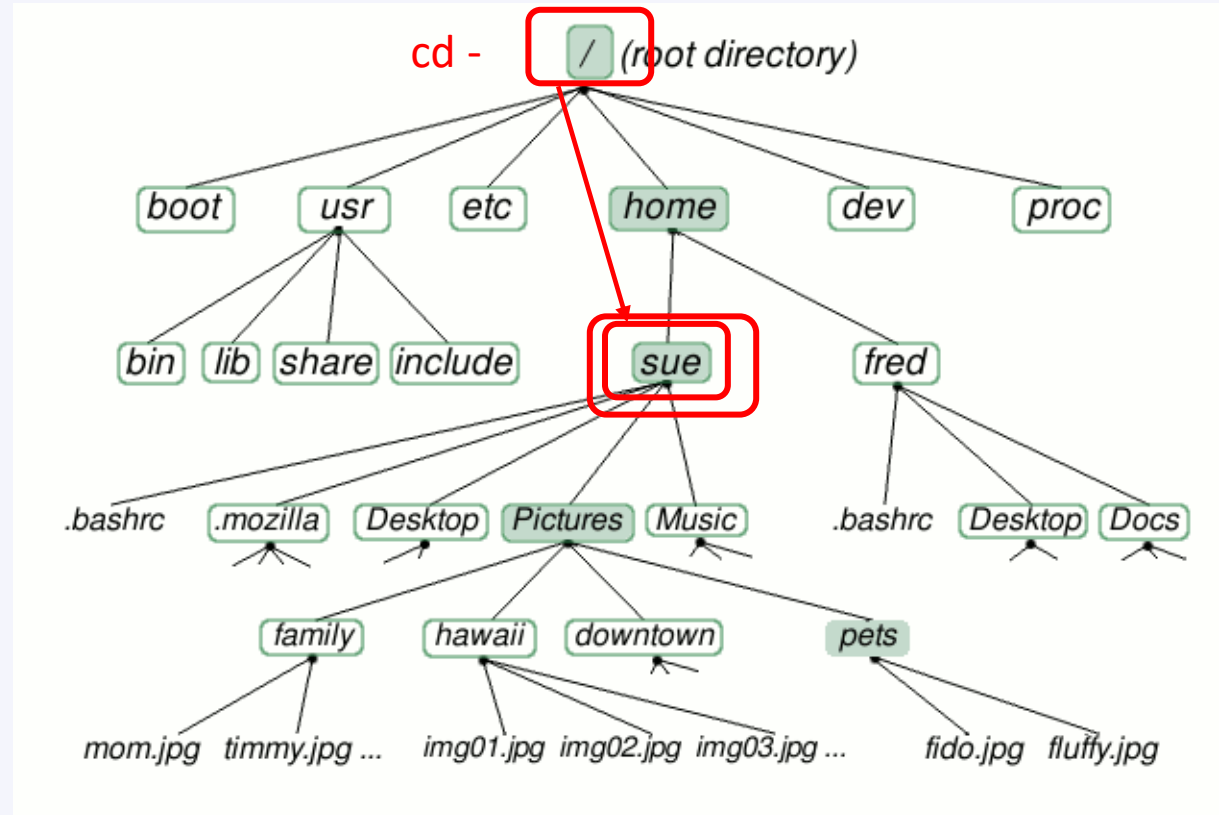
Source: <http://www.linuxplanet.com/linuxplanet/tutorials/6666/1/screenshot3894/>

# Linux File System - directories



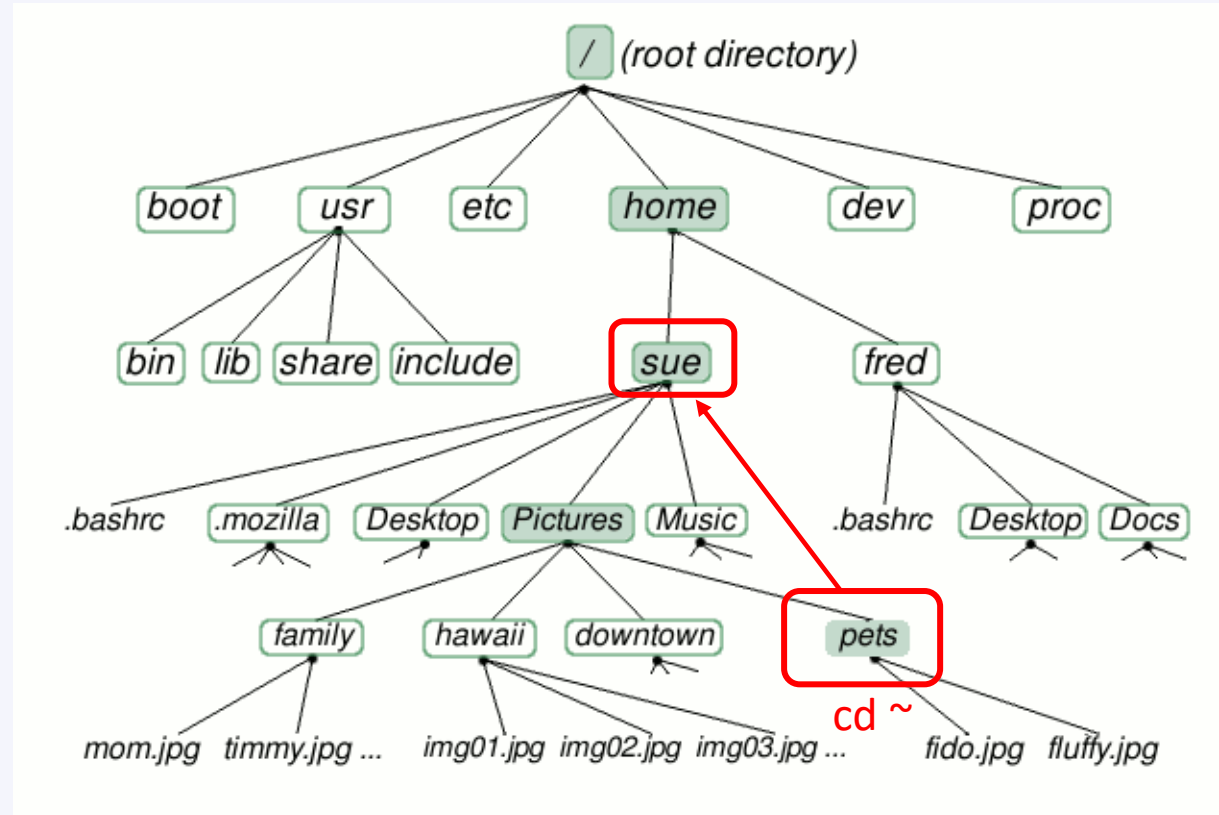
Source: <http://www.linuxplanet.com/linuxplanet/tutorials/6666/1/screenshot3894/>

# Linux File System - directories



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

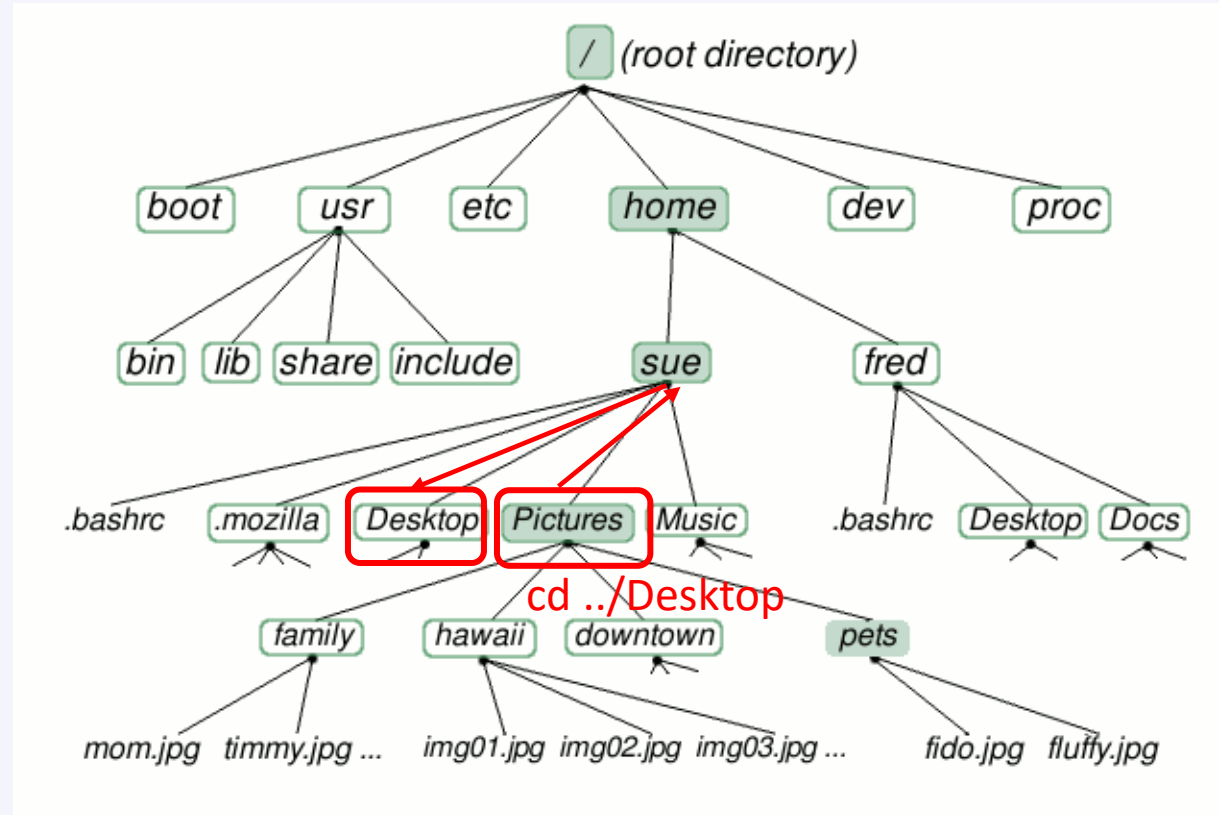
# Linux File System - directories



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

# Linux File System-home directory

Relative path

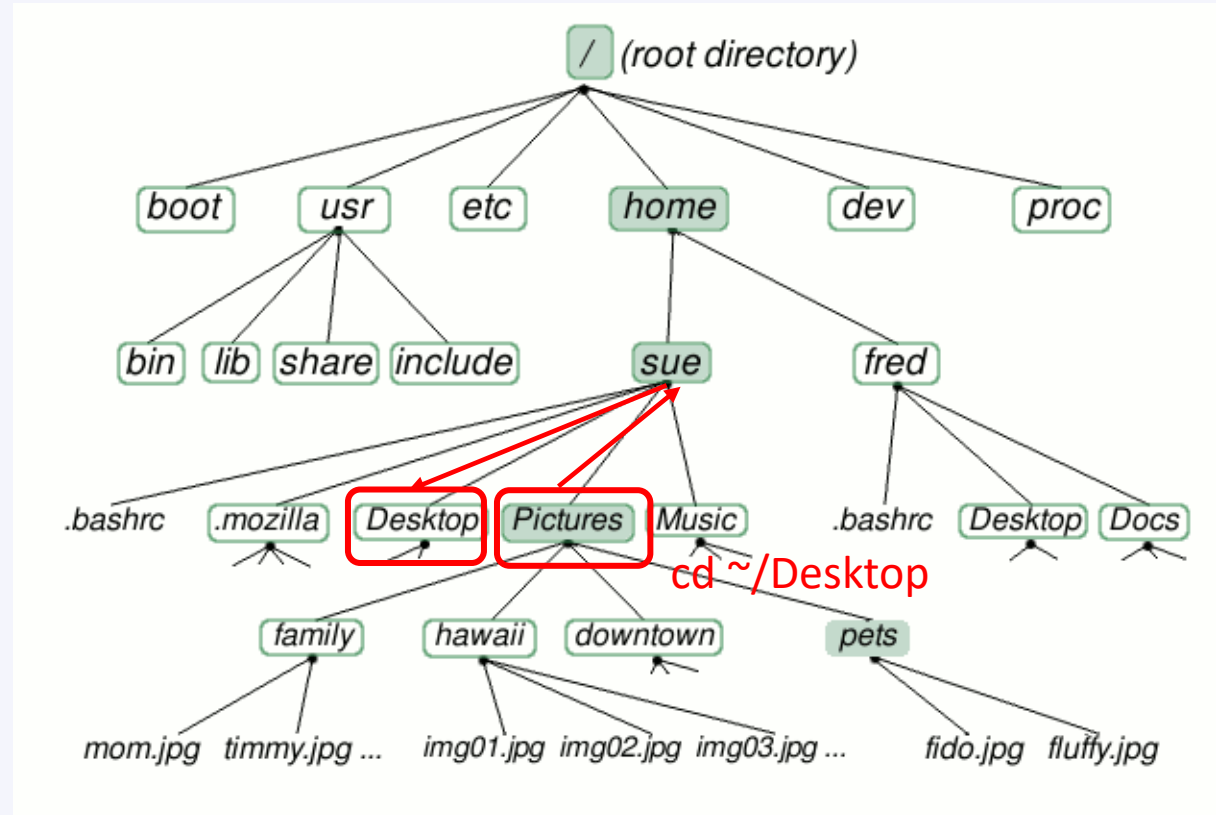


Source: <http://www.linuxplanet.com/linuxplanet/tutorials/6666/1/screenshot3894/>



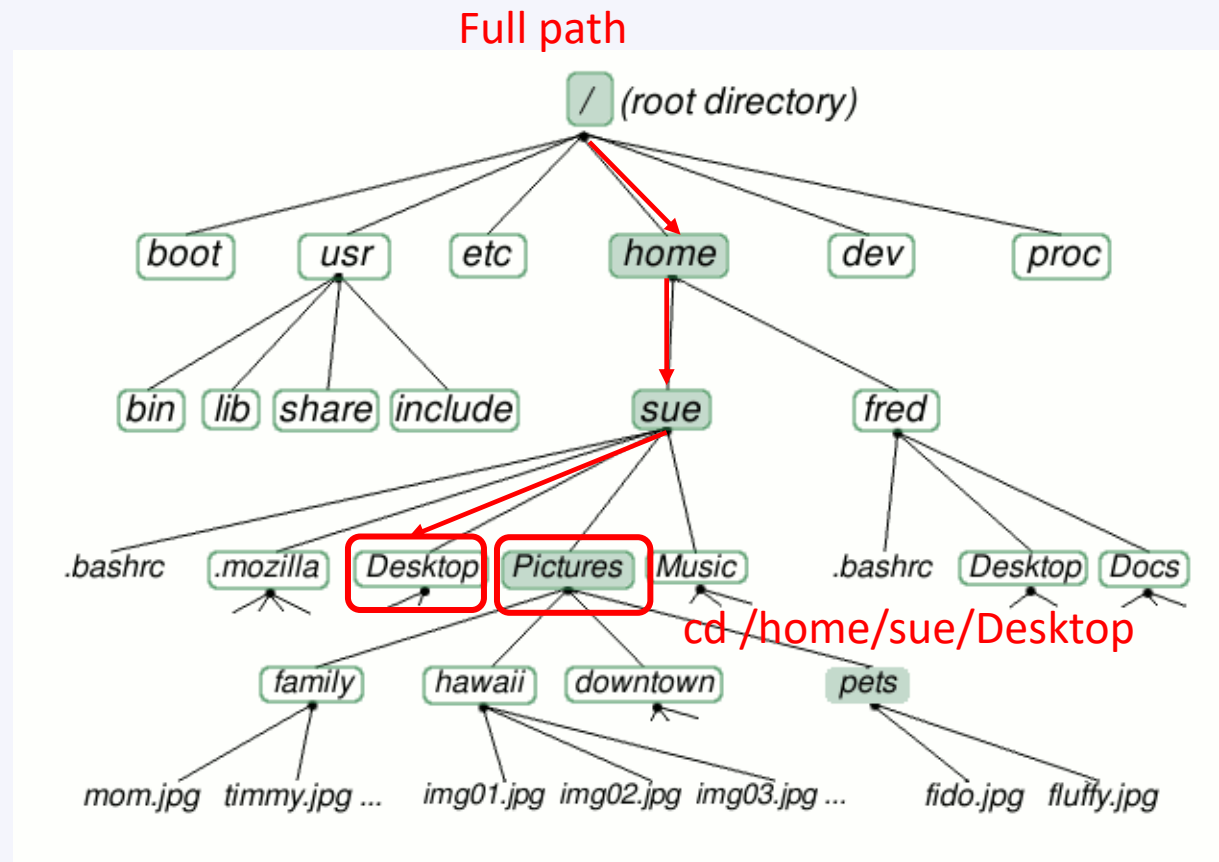
# Linux File System - directories

Relative path



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>`

# 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.



# Globbering: 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

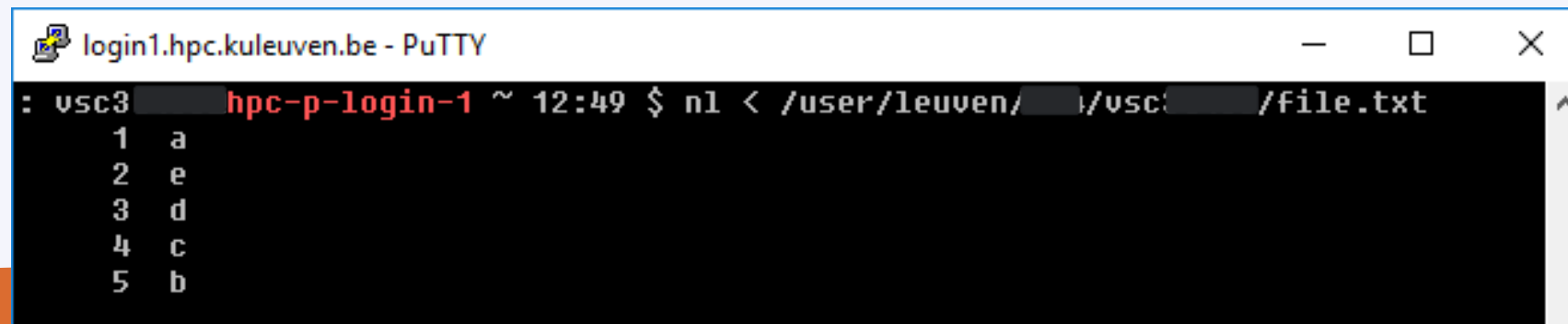
- Programs and commands can contain an input and output. These are called 'streams'. UNIX programming is oftentimes stream based.
- STDIN – 'standard input,' or input from the keyboard
- SDTOUT – 'standard output,' or output to the screen
- STDERR – 'standard error,' error output which is sent to the screen.

# 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, ...)



The screenshot shows a PuTTY terminal window titled 'login1.hpc.kuleuven.be - PuTTY'. The prompt is ': vsc3 hpc-p-login-1 ~ 12:49 \$'. The command entered is 'nl < /user/leuven/ /vsc: /file.txt'. The output of the command is a list of numbers 1 through 5, each followed by a letter: 1 a, 2 e, 3 d, 4 c, 5 b.

```
login1.hpc.kuleuven.be - PuTTY
: vsc3 hpc-p-login-1 ~ 12:49 $ nl < /user/leuven/ /vsc: /file.txt
1 a
2 e
3 d
4 c
5 b
```

# Redirecting stderr

- Performing a normal redirection will not redirect stderr. 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*

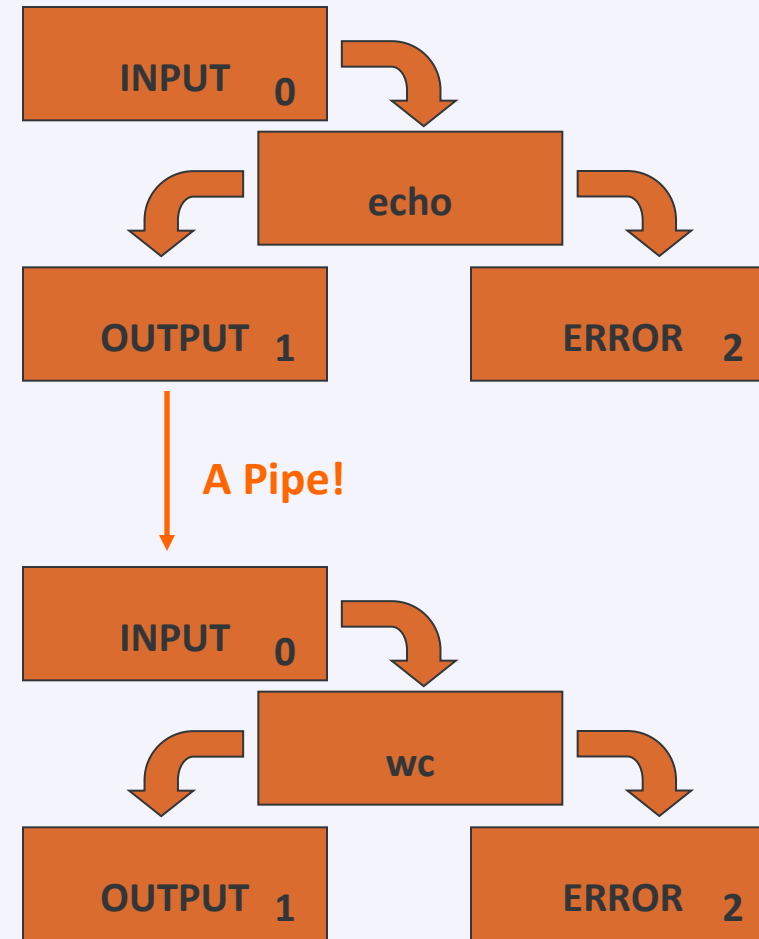
# Pipes

- 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

# 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

# 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.

# 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 exists, 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`.



# 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/>

# 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](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)

# Nano

u0076109@hpc-p-svcs-10:~

GNU nano 2.2.6 File: testfile

[ New File ]

**Exit=Ctrl+X**

**Exit** ^G Get Help ^O WriteOut ^R Read File ^Y Prev Page ^K Cut Text ^C Cur Pos  
^J Justify ^W Where Is ^U Next Page ^U UnCut Text ^T To Spell



# 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)

# 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/>

# 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.

# 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/>



# 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`

# 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
  - read permission
  - write permission
  - execute permission

# File access rights

## 3 types of **access rights**

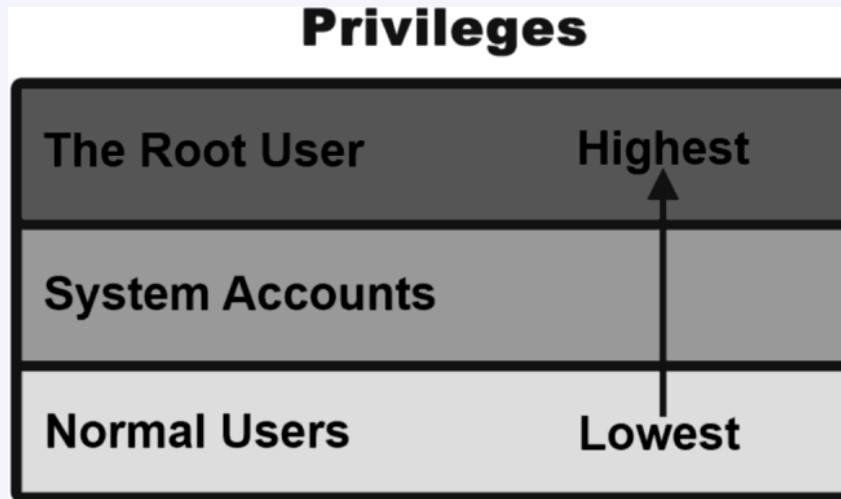
- Read access (r)
  - reading, opening, viewing, and copying the file is allowed
- Write access (w)
  - writing, changing, deleting, and saving the file is allowed
- Execute rights (x)
  - executing and invoking the file is allowed. This is required for directories to allow searching and access.

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

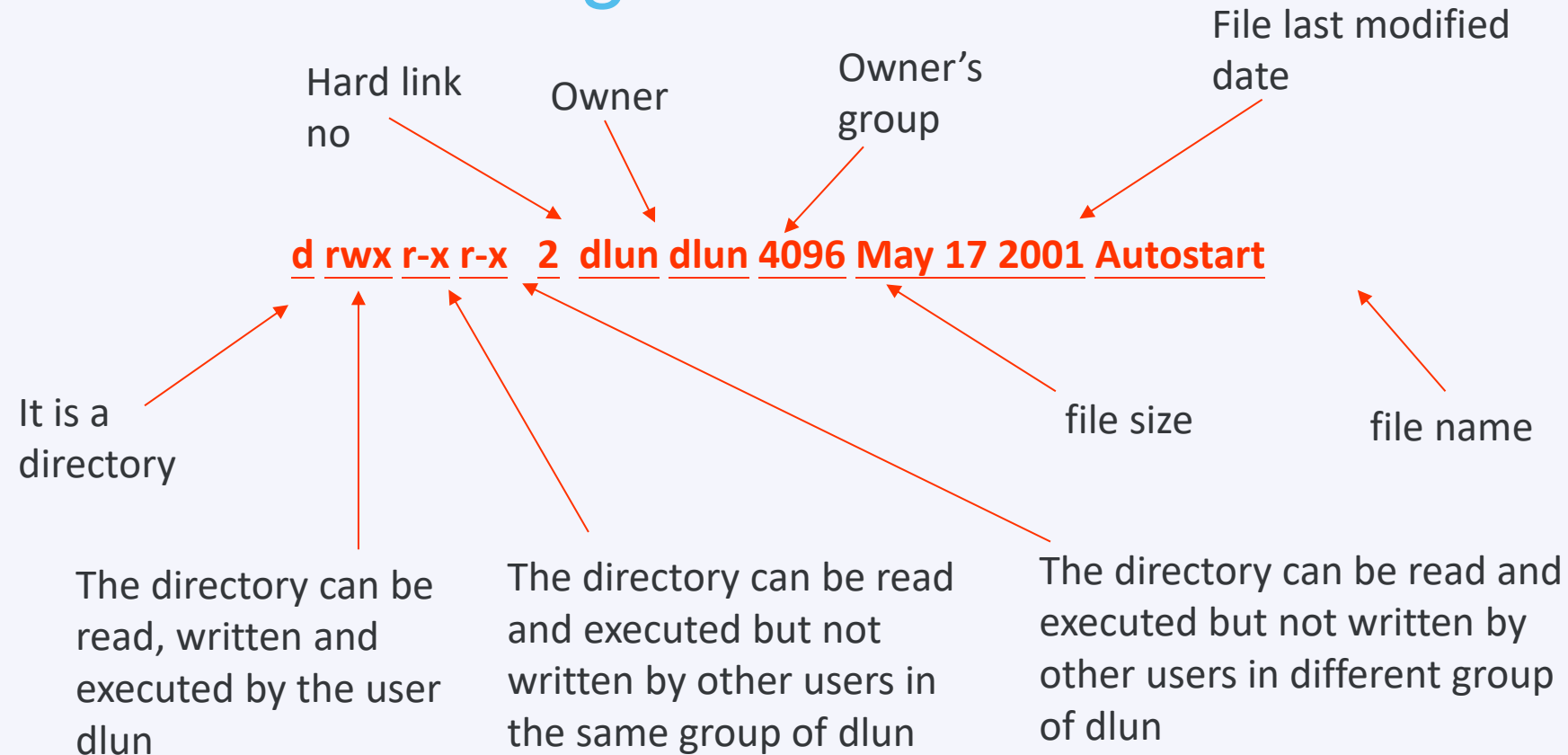
# File access rights

## 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



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

# 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...

# Access rights examples

```
dlun@enpklun.polyu.edu.hk: /home/dlun/Desktop/test/temp
File Edit Settings Help
[dlun@enpklun test]$ ls -l
total 12
-rw-r--r-- 1 dlun dlun 395 Jan 7 16:36 floppy.kde1nk
drwx----- 2 dlun dlun 4096 Jan 9 11:06 temp
-rw-rw-r-- 1 dlun dlun 16 Jan 7 16:05 test1.txt
[dlun@enpklun test]$
[dlun@enpklun test]$
[dlun@enpklun test]$ cd temp
bash: cd: temp: Permission denied
[dlun@enpklun test]$
[dlun@enpklun test]$
[dlun@enpklun test]$ chmod 700 temp
[dlun@enpklun test]$
[dlun@enpklun test]$ ls -l
total 12
-rw-r--r-- 1 dlun dlun 395 Jan 7 16:36 floppy.kde1nk
drwx----- 2 dlun dlun 4096 Jan 9 11:06 temp
-rw-rw-r-- 1 dlun dlun 16 Jan 7 16:05 test1.txt
[dlun@enpklun test]$ cd temp
[dlun@enpklun temp]$
```

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

⇒ rw- rw- ---

545 : 101 100 101

⇒ 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

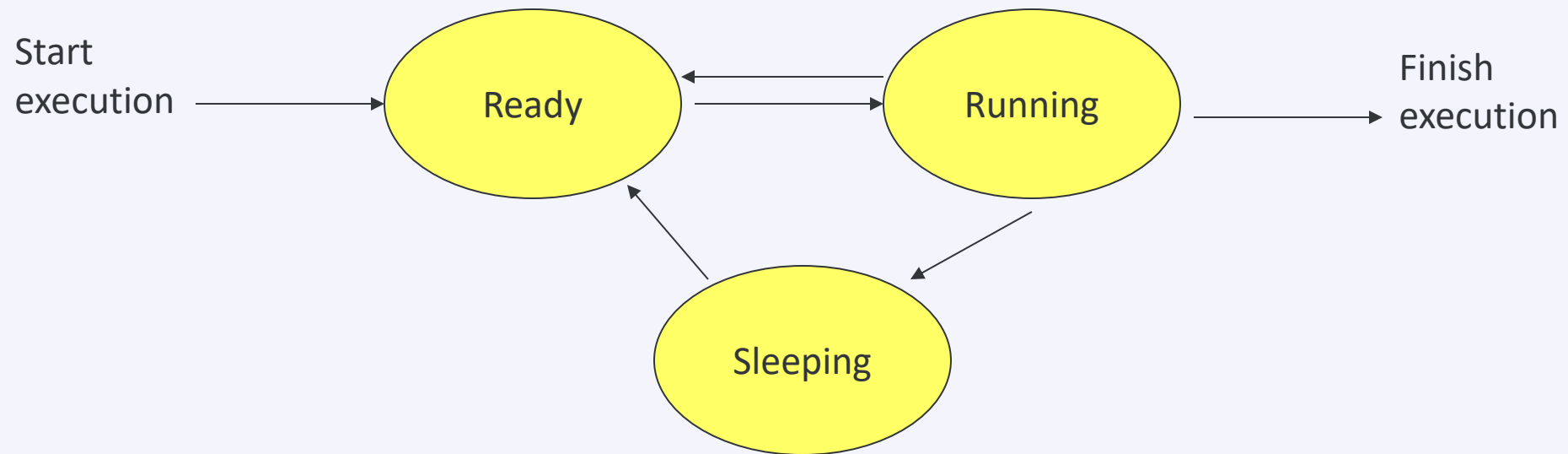
- x is sufficient to execute binaries  
Both x and r are required for shell scripts.
- Both r and x permissions are needed in practice for directories:  
r to list the contents, x to access the contents.
- You can't rename, remove, copy files in a directory if you don't have w access to this directory.
- 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.

# 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)

# 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:



# 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

# 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)



# 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

# 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.

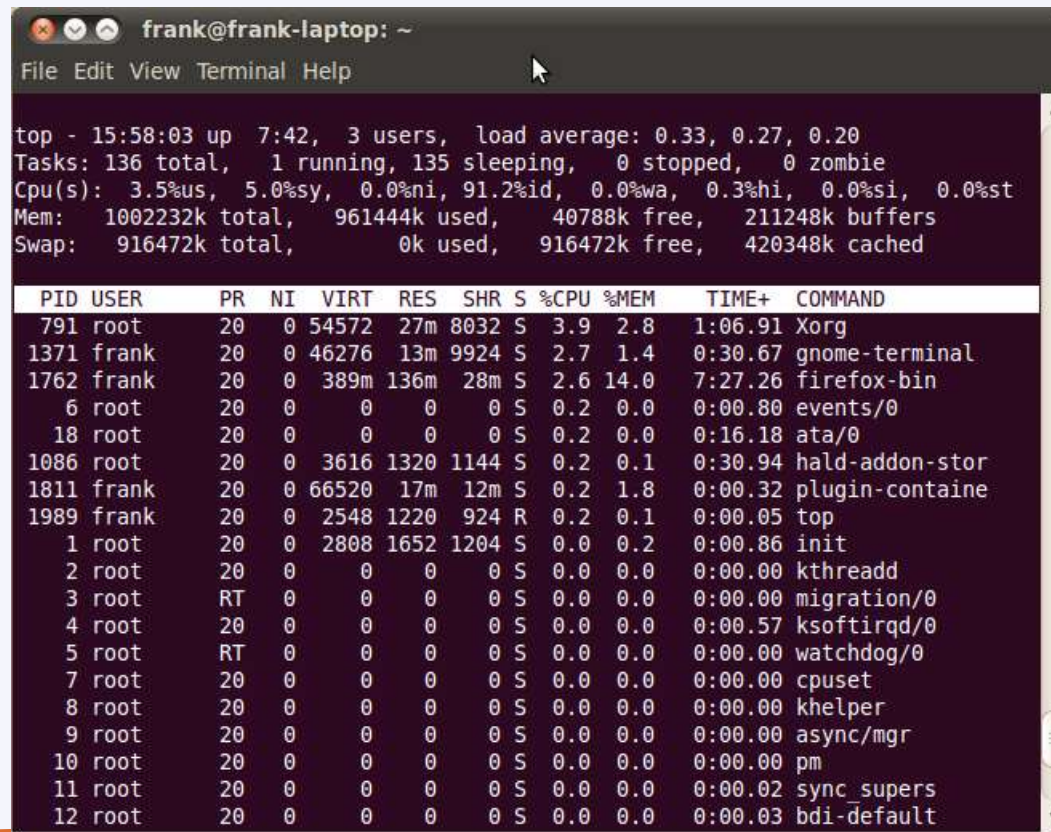
# &

- & 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

# 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/>



The screenshot shows a terminal window titled 'frank@frank-laptop: ~'. The 'top' command output is displayed, showing system statistics and a list of processes sorted by CPU usage. The processes listed include Xorg, gnome-terminal, firefox-bin, events/0, ata/0, hald-addon-stor, plugin-containe, top, init, kthreadd, migration/0, ksoftirqd/0, watchdog/0, cpuset, khelper, async/mgr, pm, sync\_supers, and bdi-default.

```
top - 15:58:03 up 7:42, 3 users, load average: 0.33, 0.27, 0.20
Tasks: 136 total, 1 running, 135 sleeping, 0 stopped, 0 zombie
Cpu(s): 3.5%us, 5.0%sy, 0.0%ni, 91.2%id, 0.0%wa, 0.3%hi, 0.0%si, 0.0%st
Mem: 1002232k total, 961444k used, 40788k free, 211248k buffers
Swap: 916472k total, 0k used, 916472k free, 420348k cached
```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
791	root	20	0	54572	27m	8032	S	3.9	2.8	1:06.91	Xorg
1371	frank	20	0	46276	13m	9924	S	2.7	1.4	0:30.67	gnome-terminal
1762	frank	20	0	389m	136m	28m	S	2.6	14.0	7:27.26	firefox-bin
6	root	20	0	0	0	0	S	0.2	0.0	0:00.80	events/0
18	root	20	0	0	0	0	S	0.2	0.0	0:16.18	ata/0
1086	root	20	0	3616	1320	1144	S	0.2	0.1	0:30.94	hald-addon-stor
1811	frank	20	0	66520	17m	12m	S	0.2	1.8	0:00.32	plugin-containe
1989	frank	20	0	2548	1220	924	R	0.2	0.1	0:00.05	top
1	root	20	0	2808	1652	1204	S	0.0	0.2	0:00.86	init
2	root	20	0	0	0	0	S	0.0	0.0	0:00.00	kthreadd
3	root	RT	0	0	0	0	S	0.0	0.0	0:00.00	migration/0
4	root	20	0	0	0	0	S	0.0	0.0	0:00.57	ksoftirqd/0
5	root	RT	0	0	0	0	S	0.0	0.0	0:00.00	watchdog/0
7	root	20	0	0	0	0	S	0.0	0.0	0:00.00	cpuset
8	root	20	0	0	0	0	S	0.0	0.0	0:00.00	khelper
9	root	20	0	0	0	0	S	0.0	0.0	0:00.00	async/mgr
10	root	20	0	0	0	0	S	0.0	0.0	0:00.00	pm
11	root	20	0	0	0	0	S	0.0	0.0	0:00.02	sync_supers
12	root	20	0	0	0	0	S	0.0	0.0	0:00.03	bdi-default

# 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

# Tier-2 credit system - what

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

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

Check ICTS service catalogue :

<https://icts.kuleuven.be/sc/english/research/HPC>

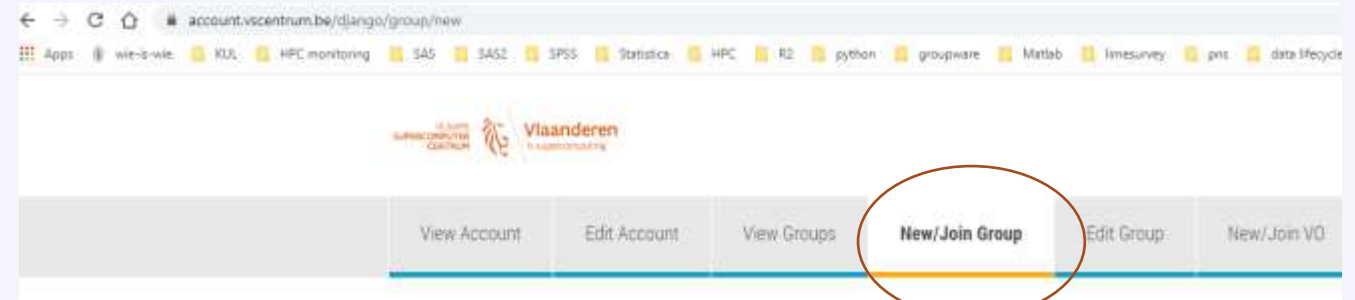
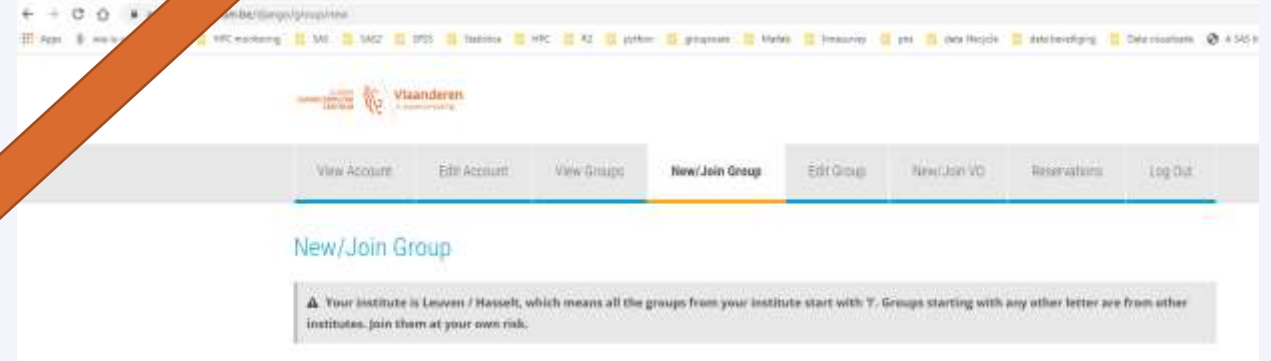
- Shared nodes are charged based on used fraction

# Tier-2 credit system – 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>

A form titled 'Create new group'. It contains a warning message: 'We will automatically prepend the letter 'T' to your groupname.' Below this is a label 'Groupname' followed by a red asterisk. There is a text input field below the label.

# 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



# Tier-2 credit system – manage your credits

- Slurm Accounting Manager => sam- commands

Command	Purpose
<code>sam-balance [-A account]</code>	List active projects and available credits

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

# Tier-2 credit system – manage your credits

- Slurm Accounting Manager => sam- commands

Command	Purpose
<code>sam-list-allocations -A &lt;account-name&gt;</code>	List the validity dates of different projects/allocations

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

# Tier-2 credit system – manage your credits

- Slurm Accounting Manager => sam- commands

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

# Tier-2 credit system – manage your credits

- Slurm Accounting Manager => sam- commands

Command	Purpose
<code>sam-statement -A &lt;account-name&gt; -s &lt;start-date&gt; -e &lt;end-date&gt;</code>	Overview of credits used on each job in a specific time window

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

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

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

# Tier-2 credit system – manage your credits

- Slurm Accounting Manager => sam- commands

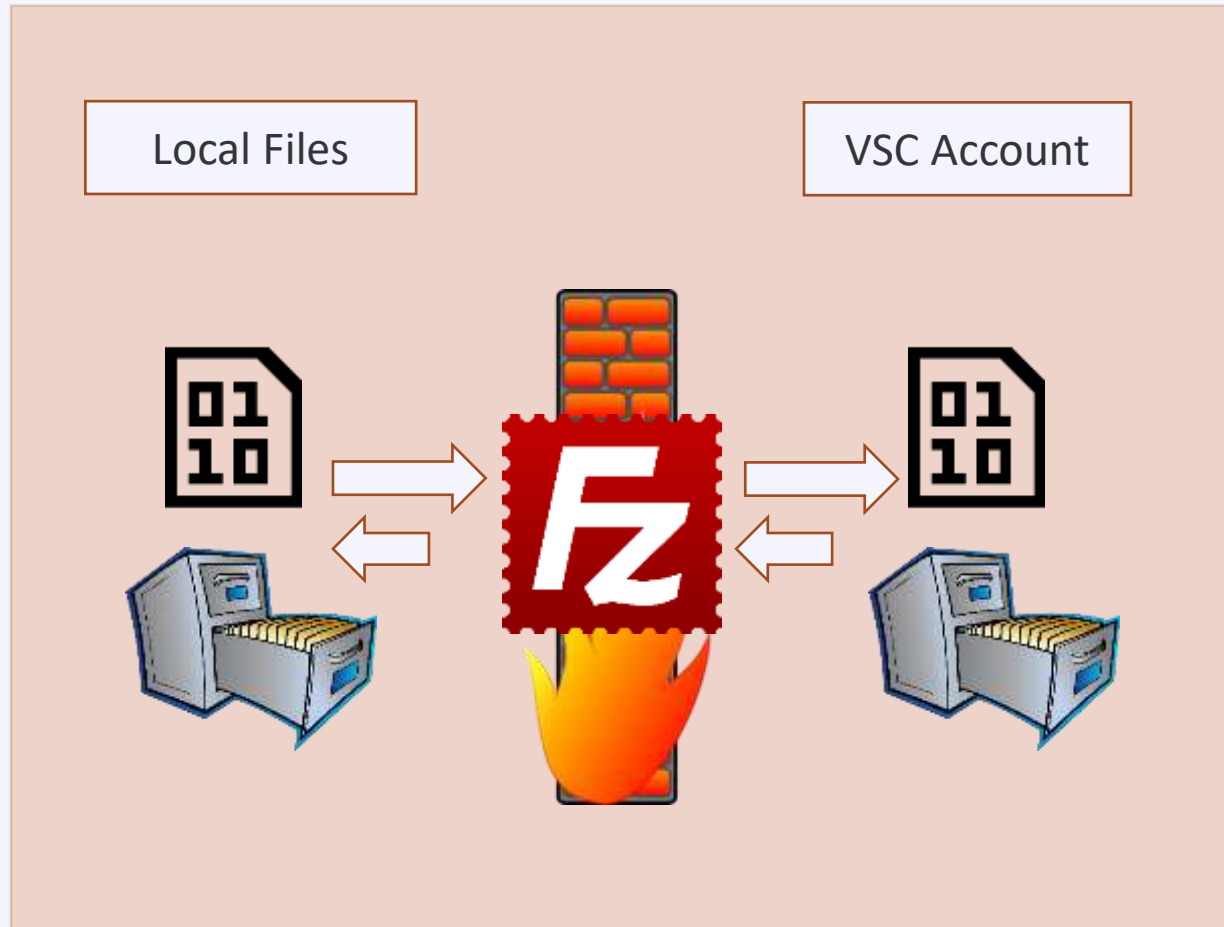
Command	Purpose
<code>sam-balance</code>	List active projects and available credits
<code>sam-list-allocations -A &lt;account-name&gt;</code>	List the validity dates of different projects/allocations
<code>sam-quote sbatch [sbatch arguments] [my_job.slurm]</code>	Shows the amount of credits a job will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



# File Transfer with FileZilla

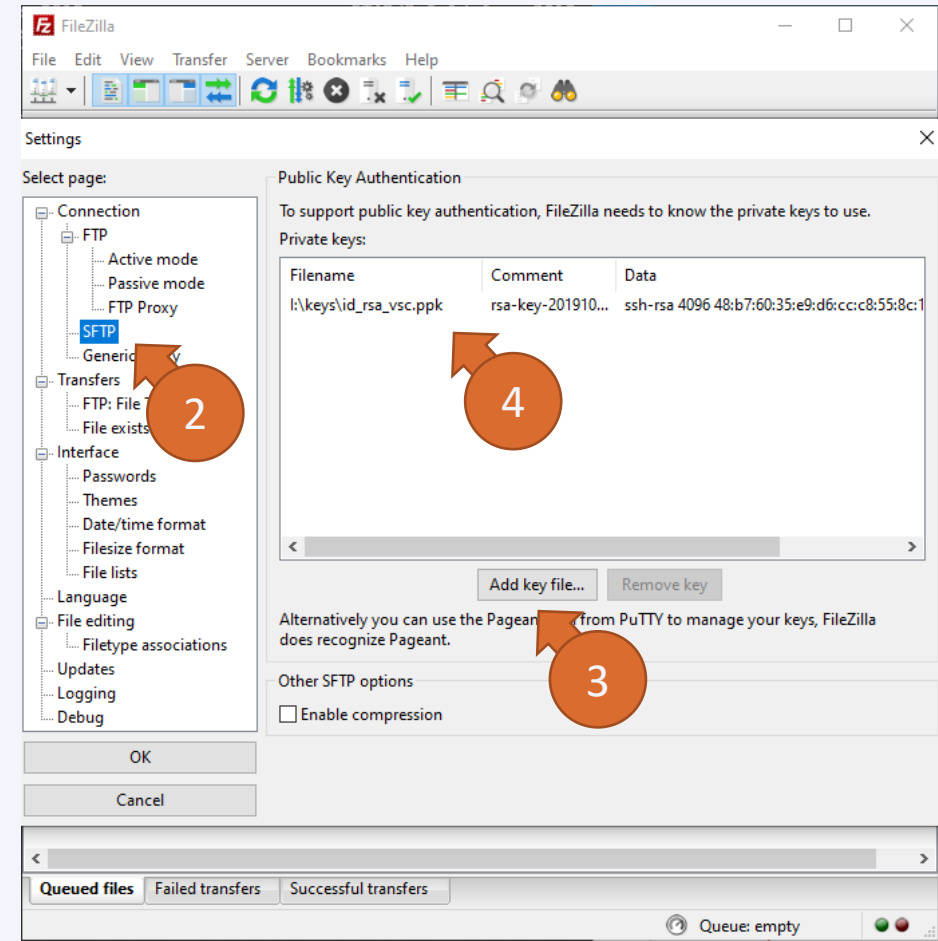
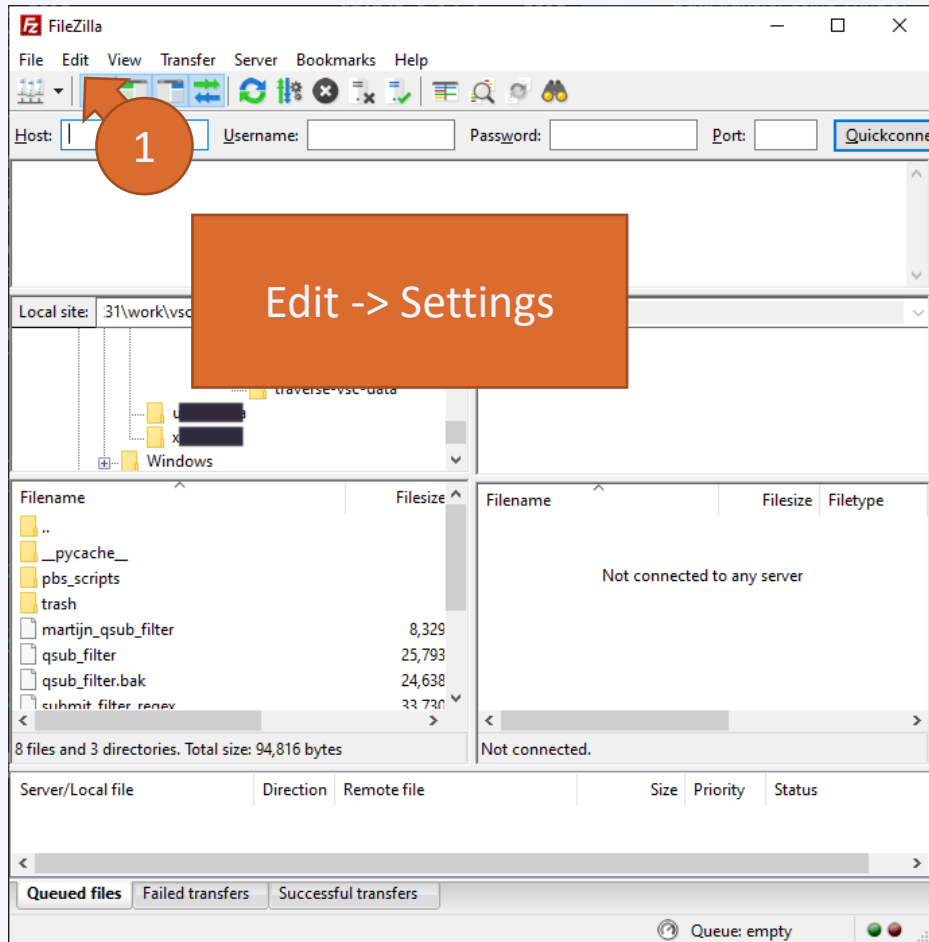


# File transfer



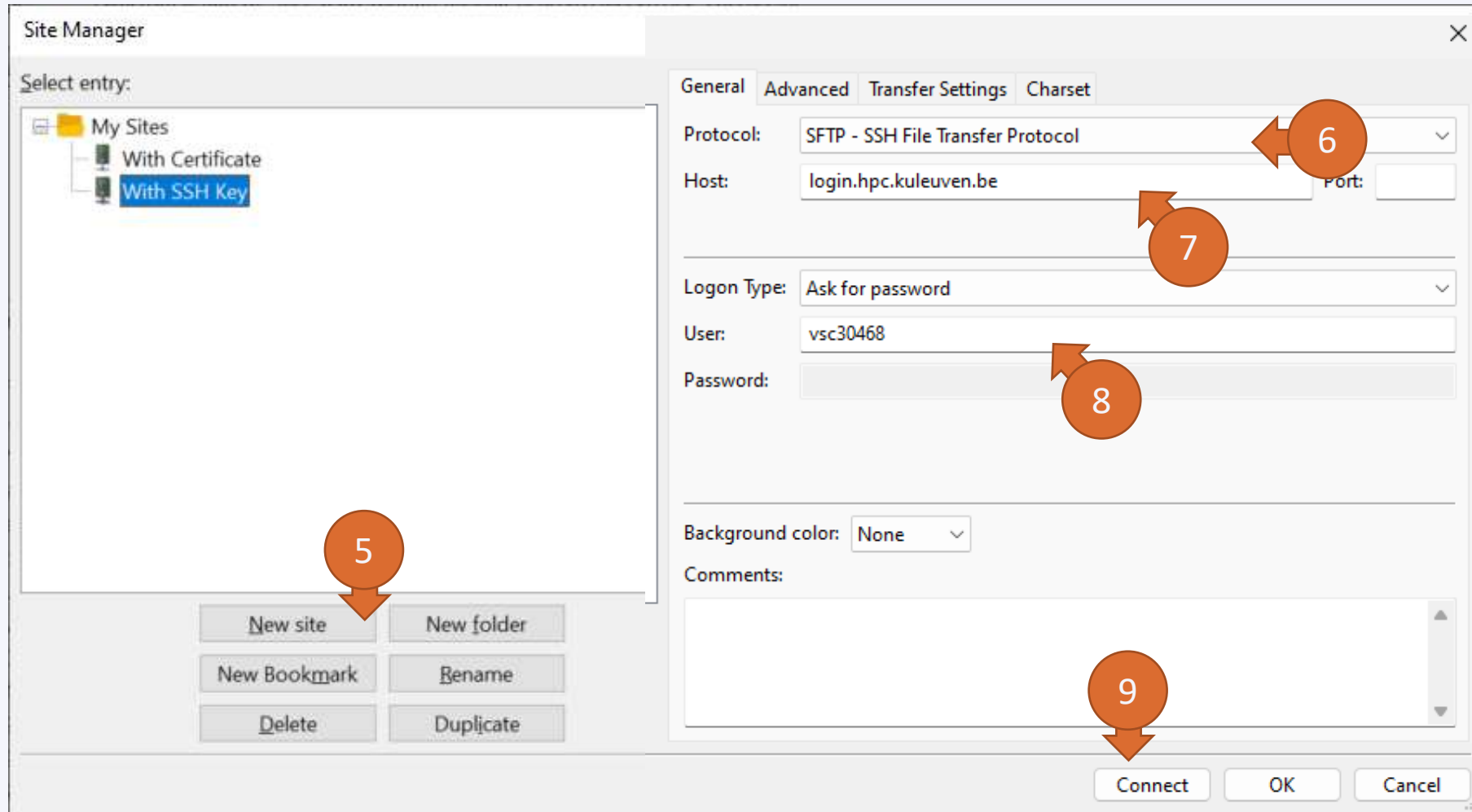
Application	OS
<a href="#">FileZilla</a>	Windows, Linux, Mac
<a href="#">WinSCP</a>	Windows
rsync, scp	Linux, Mac
<a href="#">Globus</a>	Window, Linux, Mac

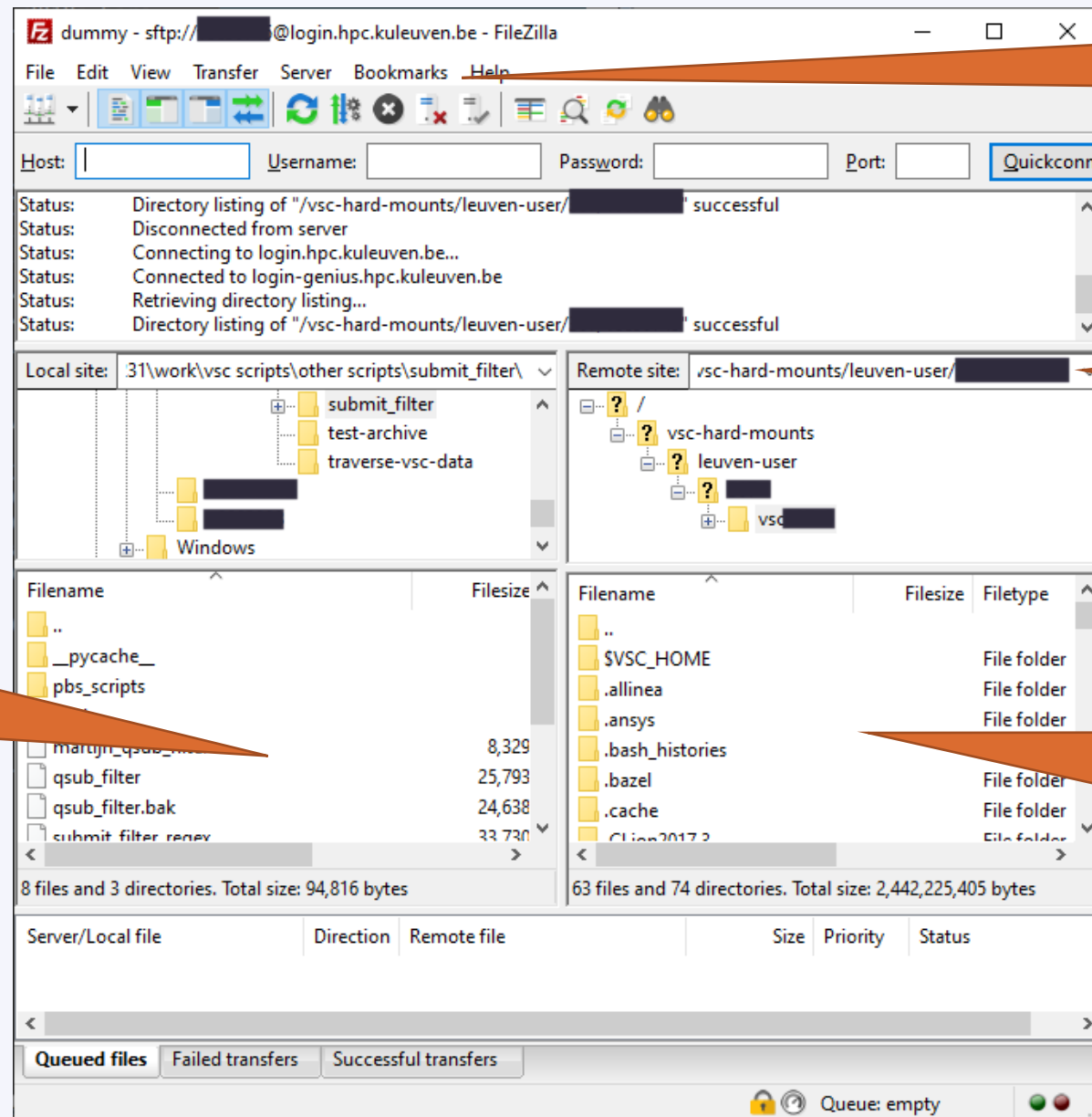
# File transfer





# File transfer





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

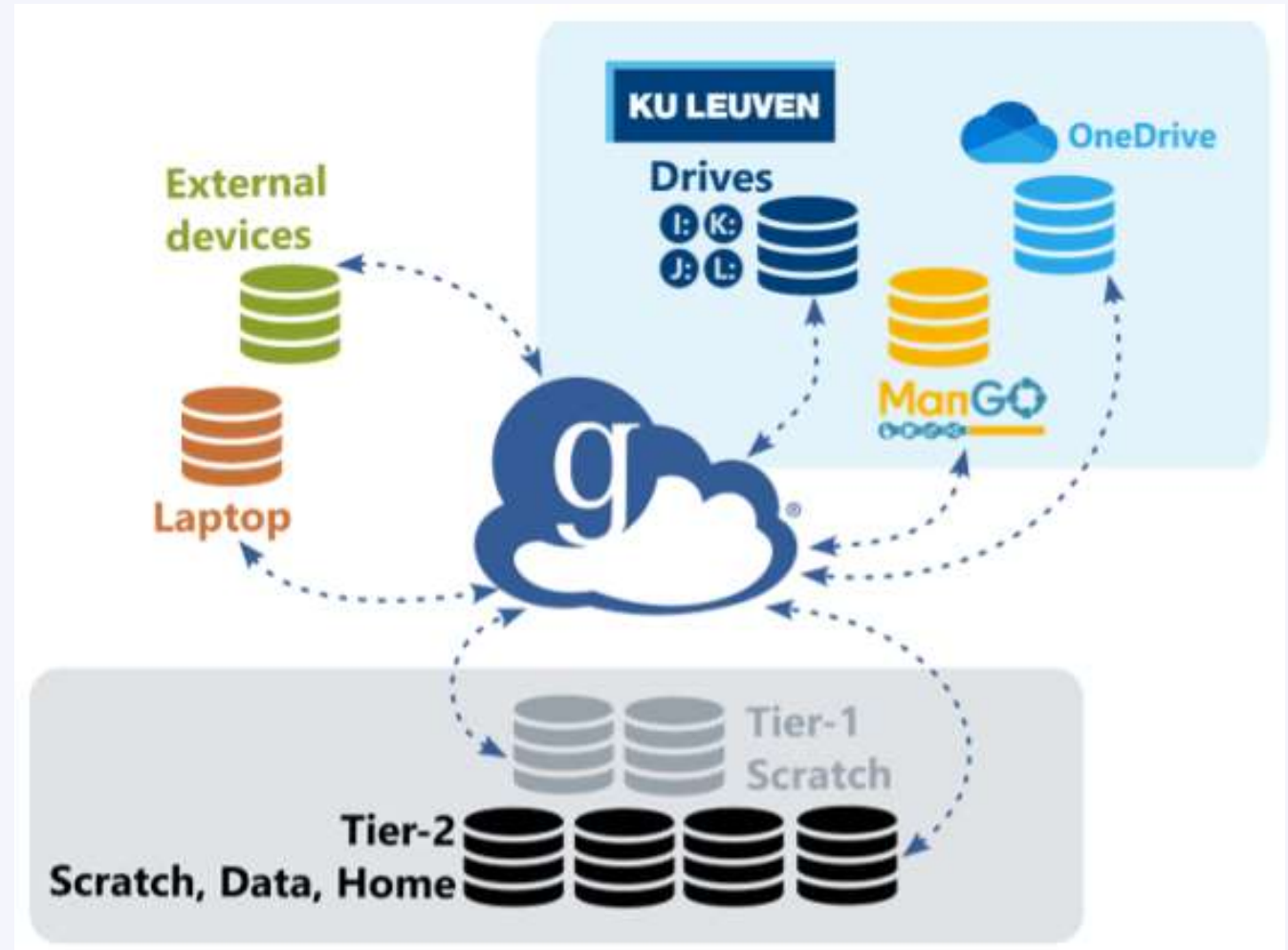
Instead, go to your \$VSC\_DATA folder, e.g. /data/leuven/399/vsc39934

Your local machine (e.g. laptop)

Your VSC storage (e.g. \$VSC\_DATA, \$VSC\_SCRATCH).  
**Note:** by default you arrive at your own \$VSC\_HOME. Copy **nothing** there!

# 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





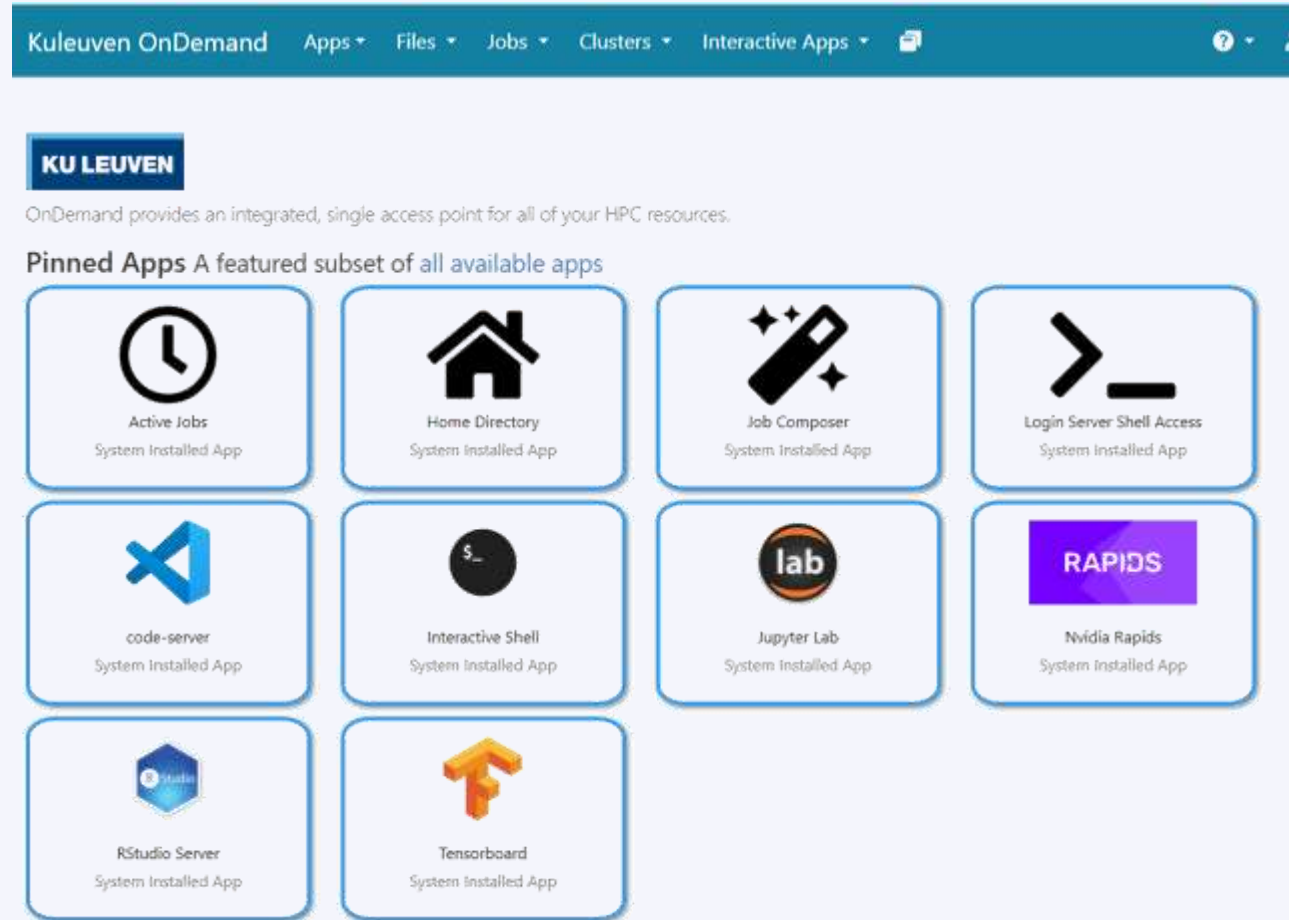
# Open Ondemand



# 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!

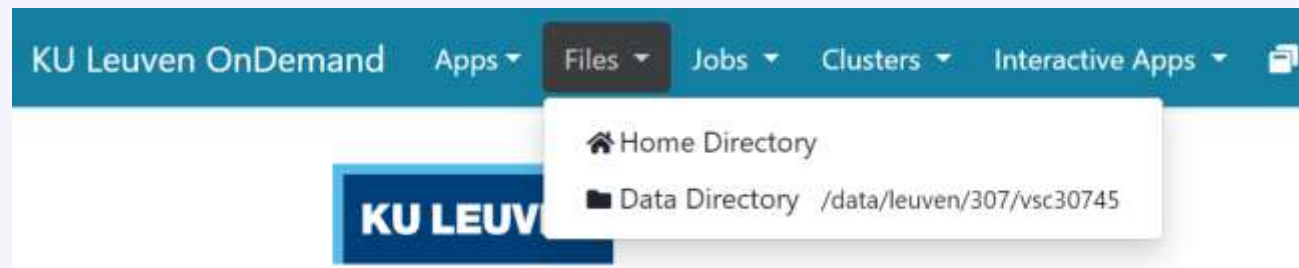


The screenshot shows the Kuleuven OnDemand web interface. At the top is a navigation bar with links: Kuleuven OnDemand, Apps, Files, Jobs, Clusters, Interactive Apps, and a help icon. Below the navigation bar is a blue header with the KULeuven logo and the text: "OnDemand provides an integrated, single access point for all of your HPC resources." Below this is a section titled "Pinned Apps A featured subset of all available apps". This section contains eight app tiles, each with an icon, a title, and the text "System Installed App". The apps are: Active Jobs (clock icon), Home Directory (house icon), Job Composer (pencil icon), Login Server Shell Access (terminal icon), code-server (VS Code icon), Interactive Shell (terminal icon), Jupyter Lab (lab icon), RAPIDS (purple box with text), RStudio Server (RStudio icon), and Tensorboard (TensorFlow icon).

# Open OnDemand

## Access your VSC storage

- ✓ Use “Files” menu to access your VSC\_HOME and VSC\_DATA
- ✓ VSC\_SCRATCH is not available



## File/Folder Management



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

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

Active Jobs  
Job Composer

Your Jobs All Clusters

Active Jobs

Show 50 entries

Filter:

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



# Open OnDemand: Login Server Shell Access

## 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.vscenrum.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 ~
$
```





# 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)

The screenshot shows the 'Interactive Apps' sidebar on the left with 'Interactive Shell' selected. The main panel is titled 'Interactive Shell' and contains the following fields:

- Cluster:** A dropdown menu with 'wice' selected.
- Account:** A dropdown menu with 'lpt2\_sysadmin' selected.
- Partition:** A text input field containing 'interactive'.
- Number of hours:** A text input field containing '1'.
- Number of cores:** A text input field containing '1'.
- Required memory per core in megabytes:** A text input field containing '3400'.
- Number of nodes:** A text input field containing '1'.
- Number of gpu's:** A text input field containing '0'.
- Reservation (optional):** A text input field.
- Name of an existing reservation in which the job should run:** A text input field.
- Launch:** A blue button.

Below the 'Launch' button, there is a note: '\* The Interactive Shell session data for this session can be accessed under the data root directory.'



# Open OnDemand: Jupyter Lab

## Jupyter Lab

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

**Interactive Apps**

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

**Jupyter Lab**

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

Cluster:

Account:

Partition:

batch\_long or bigmem or interactive or gpu\_p100\_v100 or dedicated...

☐ Enable mldlr rapide

Only available on wCE interactive or gpu nodes with a GPU requested.

Number of hours:

Number of cores:

Required memory per core in megabytes:

Number of nodes:

Number of gpus:

(Optional) Specify the total number of GPUs slots for the job. An optional GPU type specification can be supplied. For example "A100" or "T4".

Reservation (optional):

Name of an existing reservation in which the job should run

☐ I would like to receive an email when the session starts

Pre-run scriptlet:

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

\* The Jupyter Lab session data for this session can be accessed under the data root directory.



# Open OnDemand: Custom R/Python Kernels

My Kernels

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

## Step 1: Miniconda

Start an Interactive shell

Go to your data folder

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

Download miniconda

```
$ wget https://repo.continuum.io/miniconda/Miniconda3-latest-Linux-x86\_64.sh
```

Install miniconda into your data folder

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

Permanently add the path to miniconda to your ~/.bashrc

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



# Open OnDemand: Custom R/Python Kernels

## Step 2: New Environment

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

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

Enable your new environment

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

Add the ipykernel package

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

## Step 3: Register Your Kernel

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

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



## Open OnDemand: Other tools



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.



# 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

Home / My Interactive Sessions / Jupyter Lab

Interactive Apps

Servers

- Interactive Shell
- Jupyter Lab**
- RStudio Server
- Tensorboard
- code-server
- Work in progress
- ParaViewWeb - Work in progress
- cryosparc

### Jupyter Lab

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

Account

lp\_myproject

Partition

interactive

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

Number of hours

3

Number of cores

1

Required memory per core in megabytes

3400

Number of nodes

1

Number of gpu's

0

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

Reservation (optional)

Name of an existing reservation in which the job should run

☐ I would like to receive an email when the session starts

Launch





# Software Stack

# Software: Available Modules

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

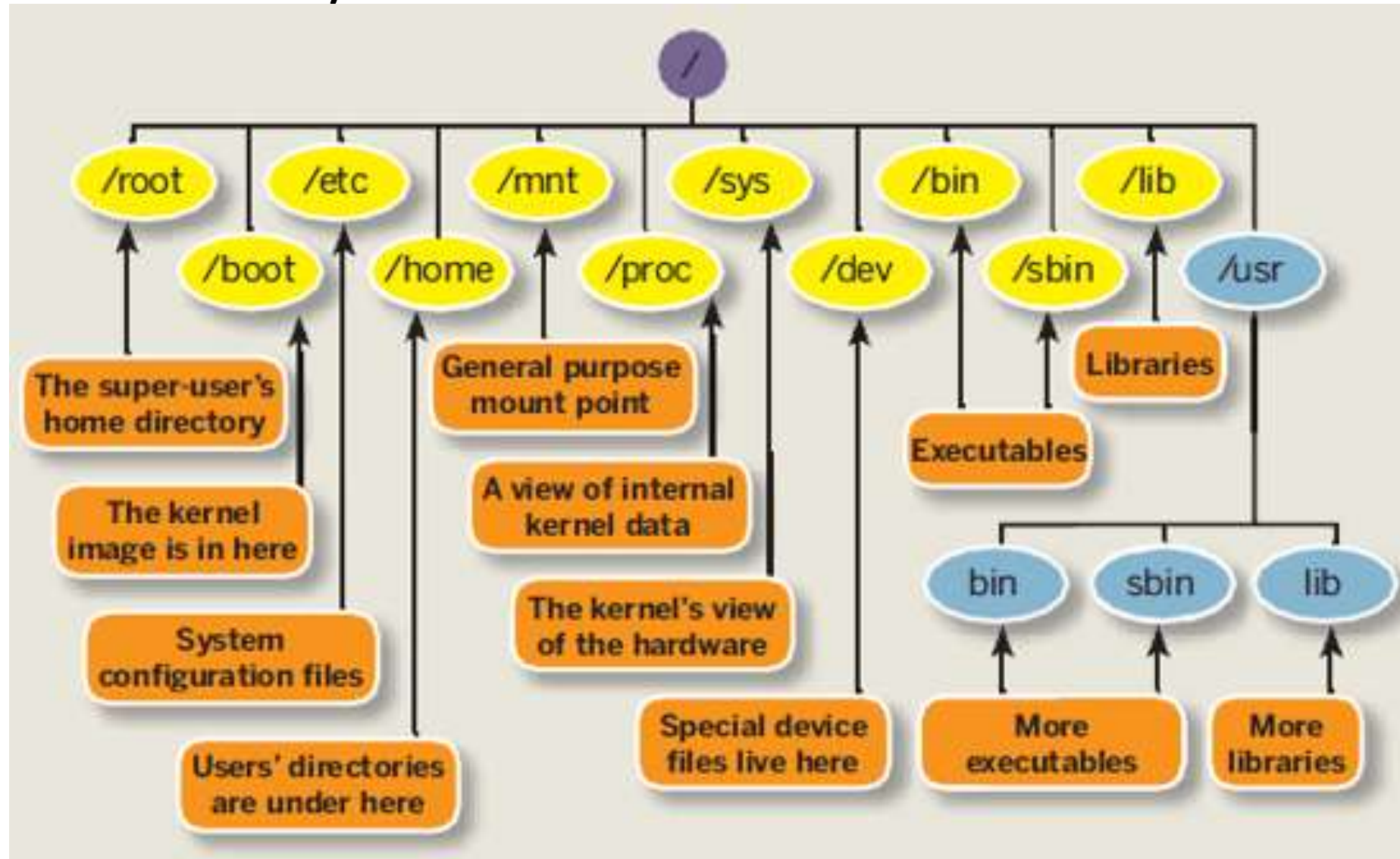
Command	Remark
<code>module av</code>	List all available modules
<code>module av Python</code>	List all Python-related modules
<code>module spider Python</code>	Get more info
<code>module load Python/3.6.4-intel-2018a</code>	Load a specific module
<code>module list</code>	List all loaded modules and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



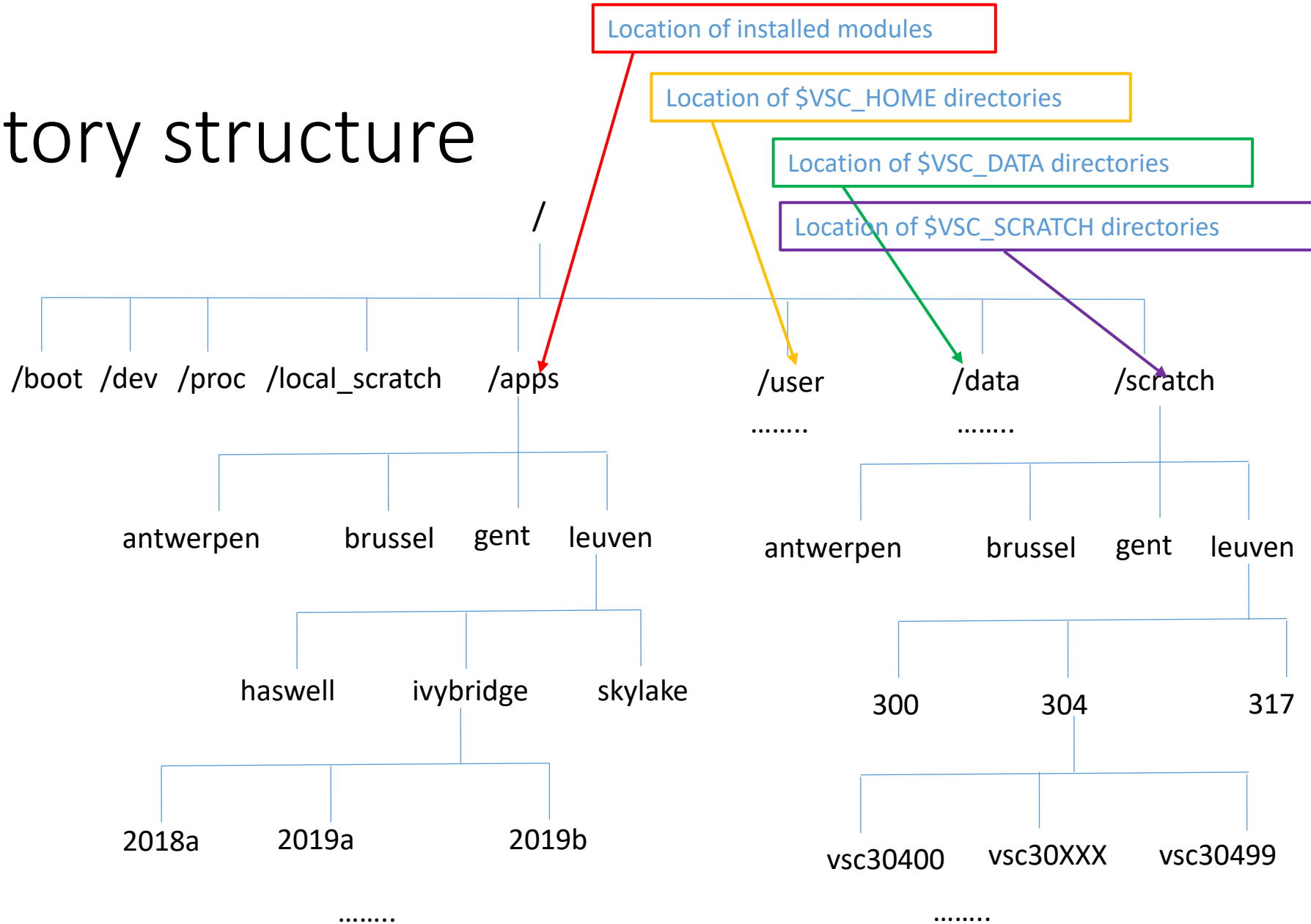
# Software: Your Specific Needs

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

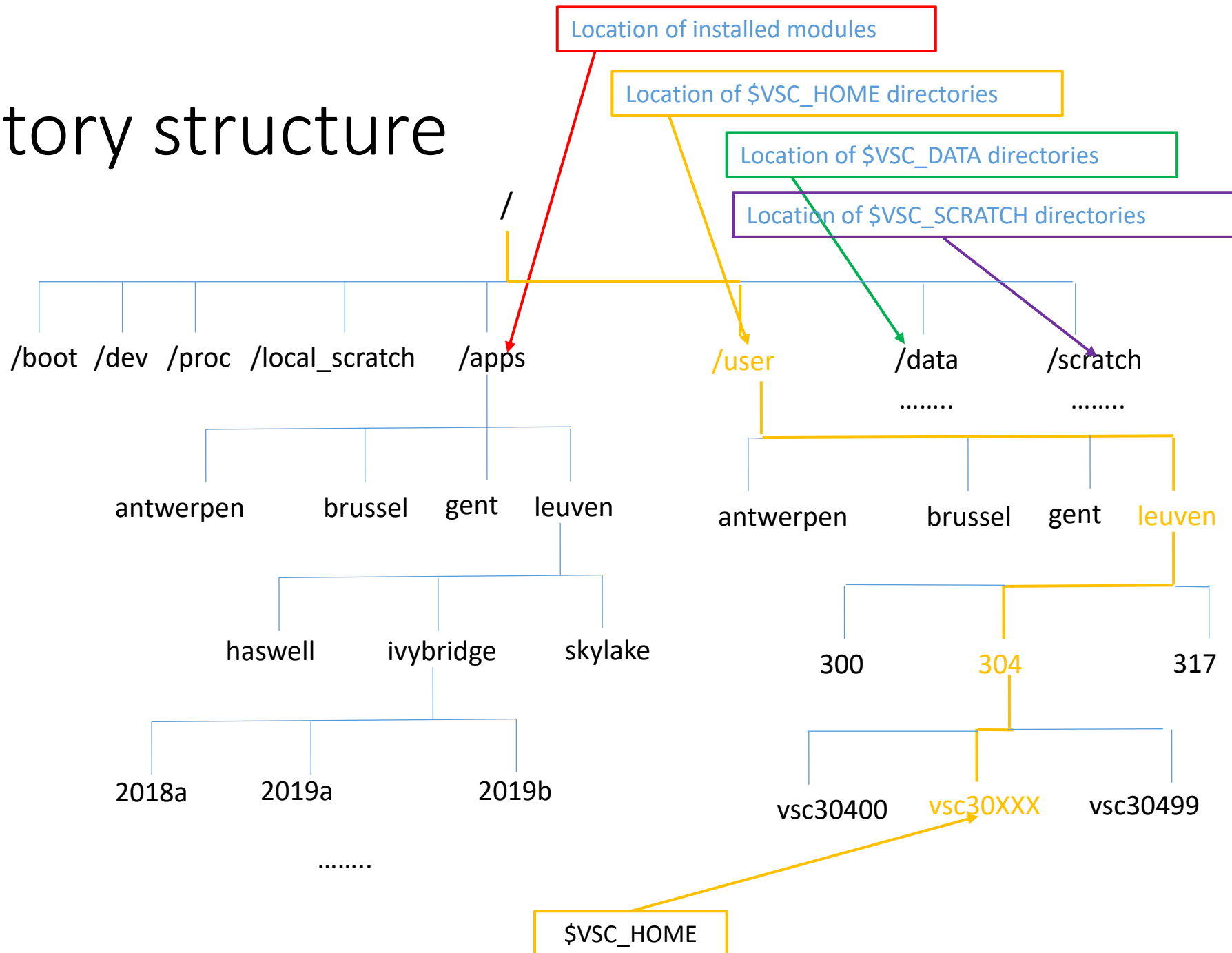
# Linux File System



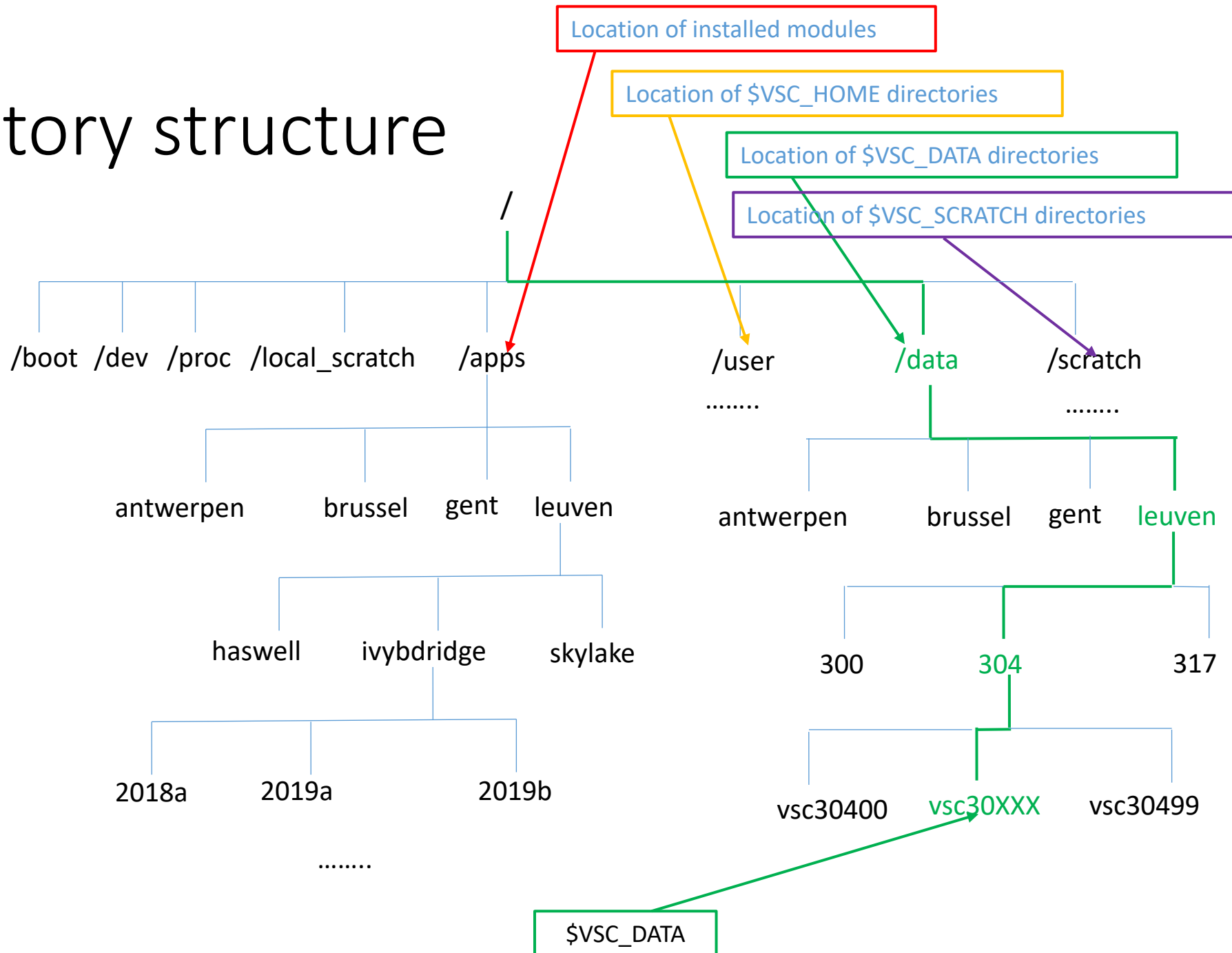
# Directory structure



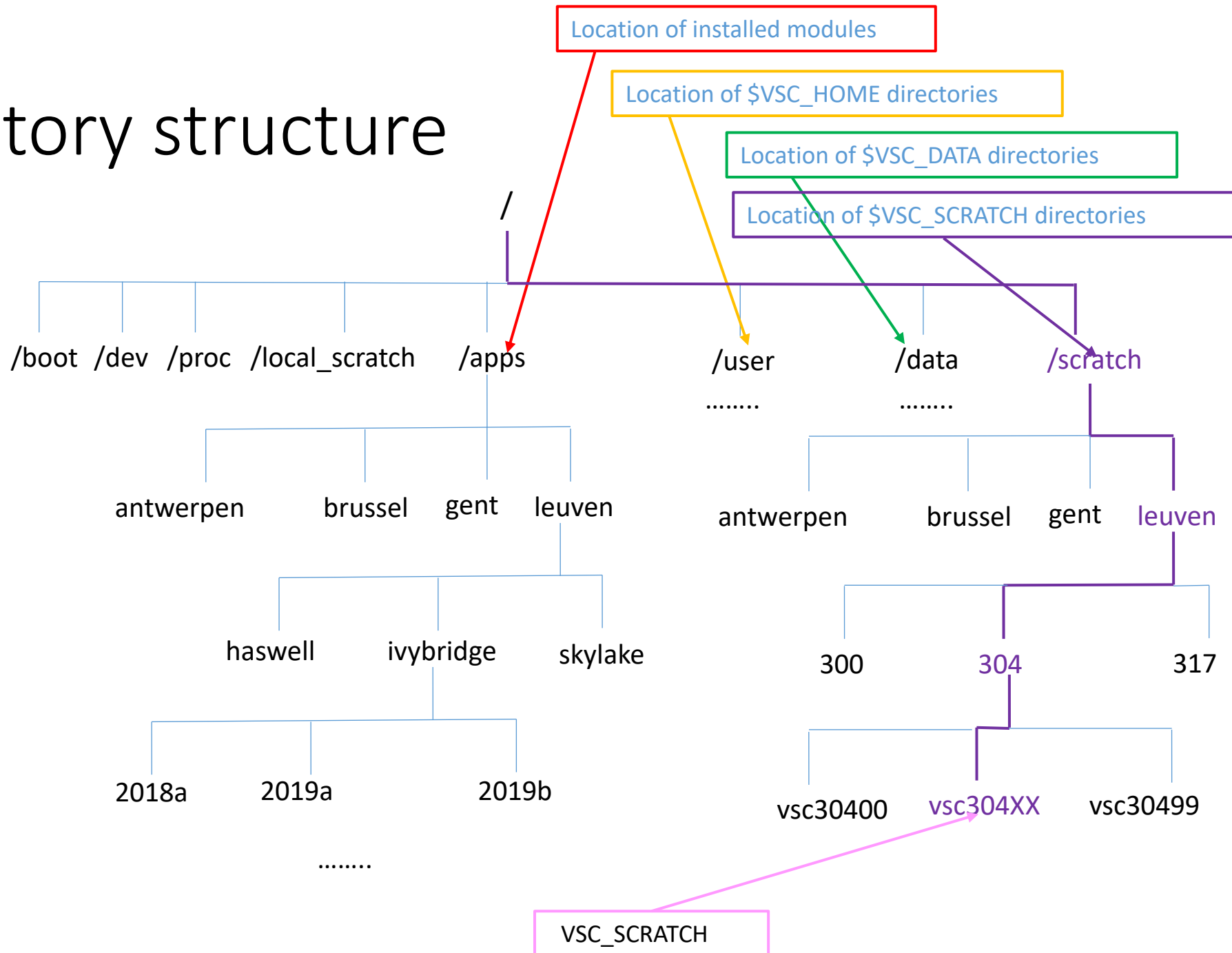
# Directory structure



# Directory structure



# Directory structure





# 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/Mail/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 version 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

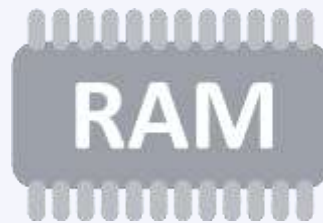
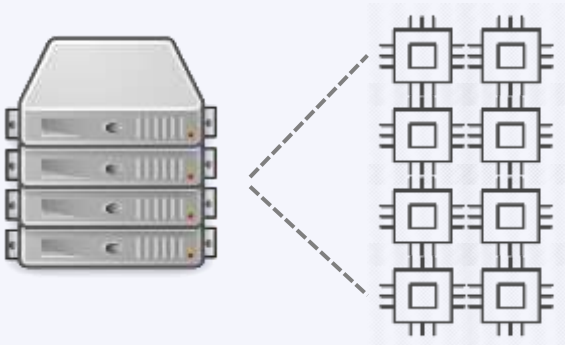
# Resource Glossary

- ✓ **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?



# Default Resources

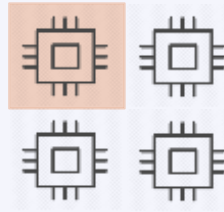
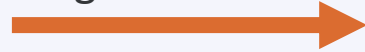
- ✓ **Cluster:** the local machine you are logged into (Genius or wICE)
- ✓ **Nodes:** 1
- ✓ **Cores:** 1
- ✓ **RAM:** depends on which node(s) you use
- ✓ **Partition:** batch
- ✓ **Walltime:** 1 hour
- ✓ **Storage:** no default
- ✓ **Credits:** no default



## Serial Application

(1 process on 1 core)

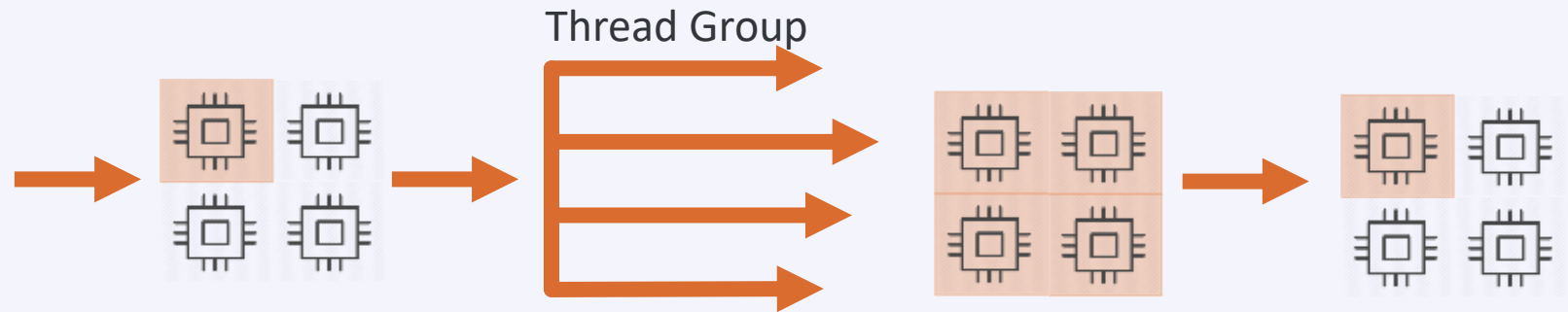
Single Process



One node

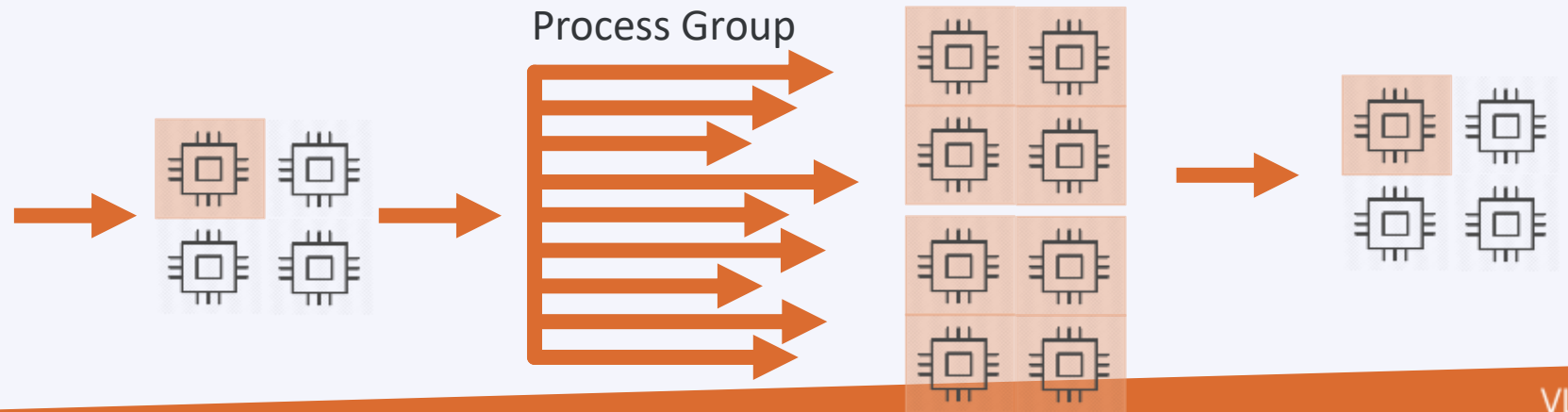
## Multi-Core Appl.

(N threads on N cores  
from **1 node**)

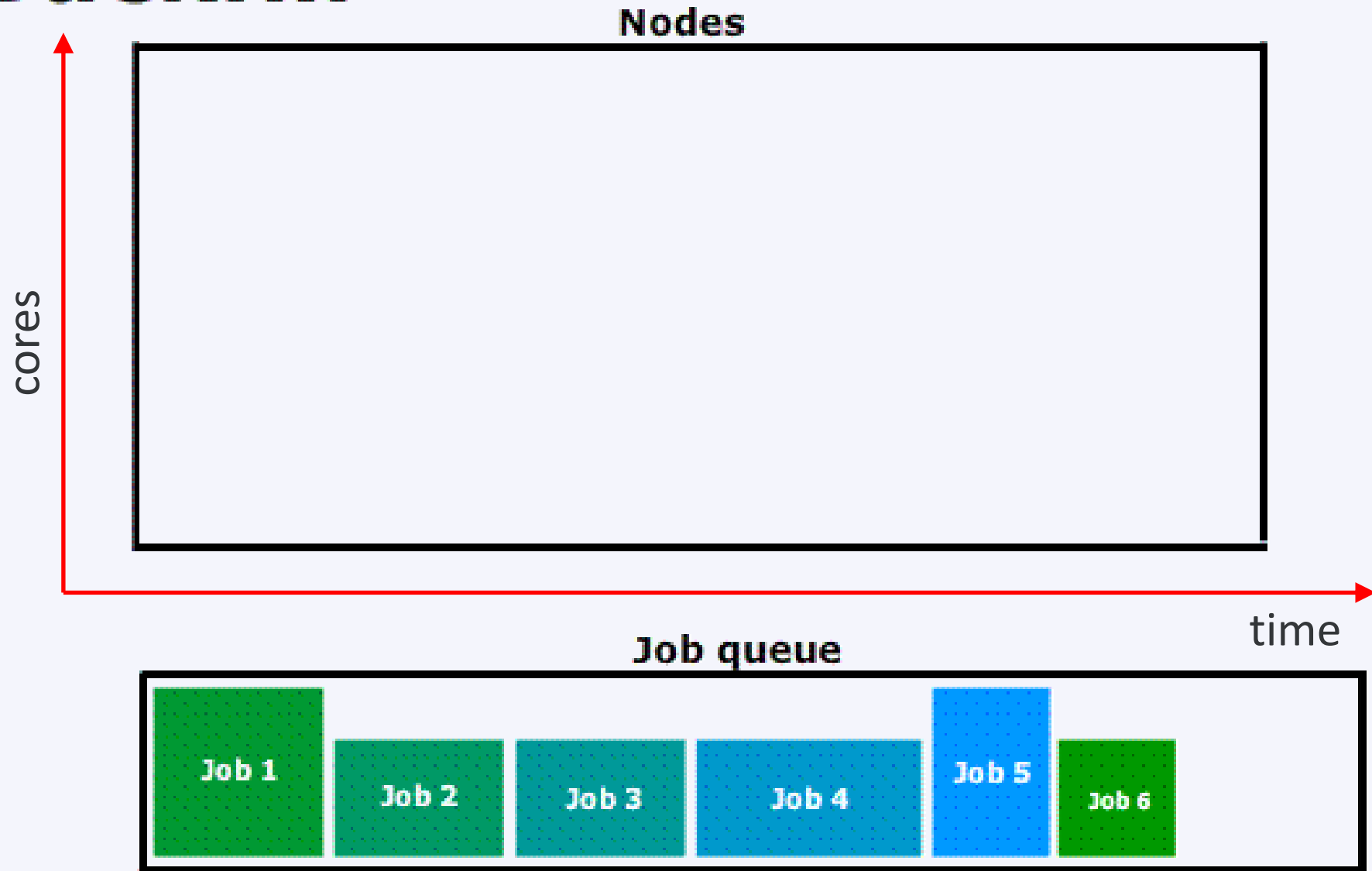


## Distributed Appl.

(many processes on  
multiple cores/nodes)

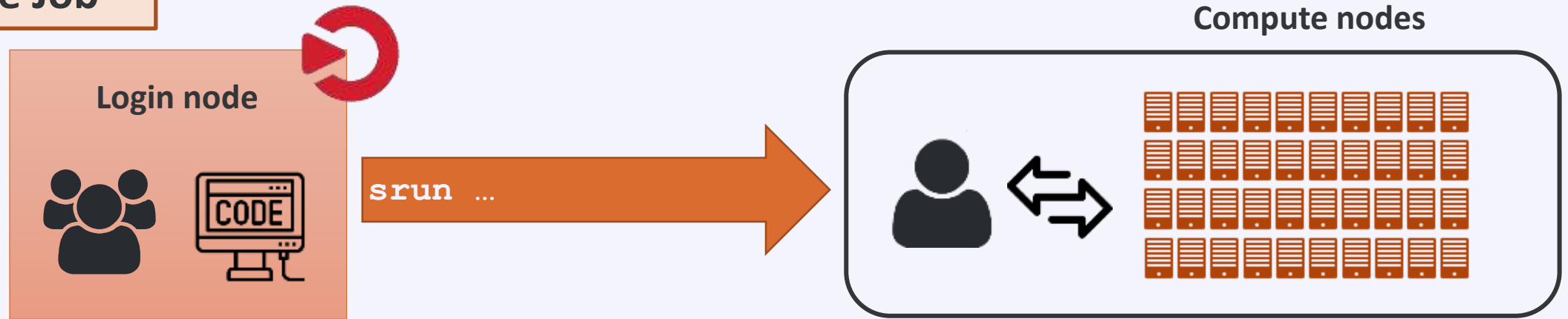


# Backfill

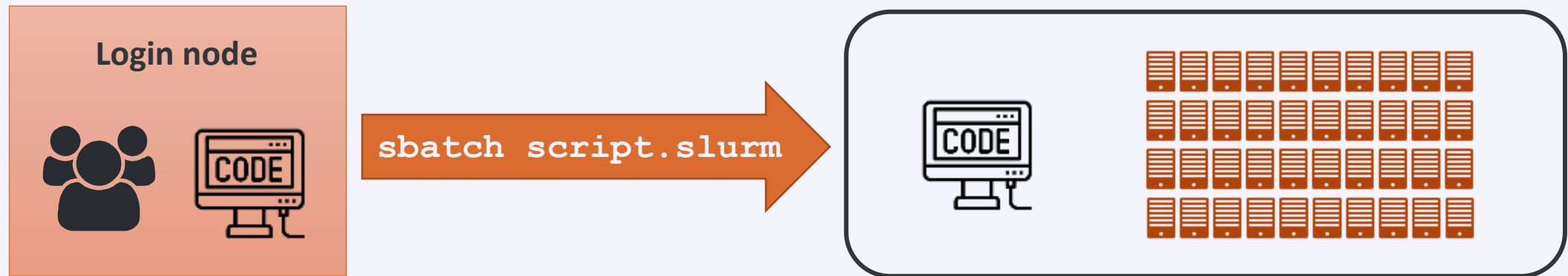




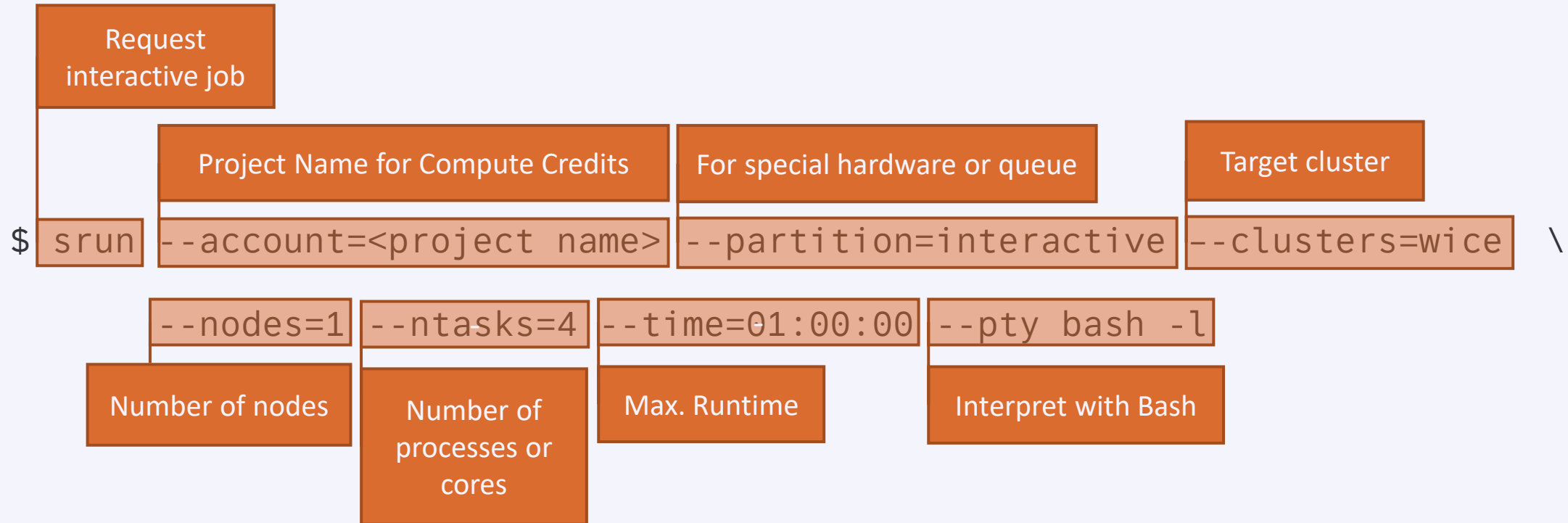
## Interactive Job



## Batch Job



# Interactive Job on **wICE**



## Remark

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



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

# Jobs: Slurm submit options

SLURM	Remarks
<code>--nodes=X --ntasks-per-node=Y</code> <code>--ntasks=X*Y</code> <code>--ntasks=X --cpus-per-task=Y</code>	or or (e.g. Hybrid MPI + OpenMP)
<code>--partition=&lt;partition_name&gt;</code>	Default: "batch" partition
<code>--mem-per-cpu=&lt;size&gt;&lt;M&gt;</code>	Min memory per core, e.g. 5000M for Genius
<code>--time=&lt;dd-hh:mm:ss&gt;</code>	e.g. 1-12:30:00
<code>--job-name=&lt;job_name&gt;</code>	
<code>--output=&lt;file_template&gt;</code>	STDOUT; default="slurm-%j.out"
<code>--error=&lt;file_template&gt;</code>	Default: redirect to STDOUT
<code>--mail-type=FAIL,BEGIN,END</code>	
<code>--mail-user=&lt;email-address&gt;</code>	
<code>--export=&lt;ALL, key=value&gt;</code>	Additional variables to pass to job
<code>--account=&lt;account_name&gt;</code>	Mandatory (credits)

# Argument Shorthands

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

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

# Interactive Jobs

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

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

- ✓ Interactive job with X-forwarding

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

- ✓ Request fraction of a node from interactive partition

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

- ✓ Request a GPU accelerator

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

- ✓ To join a running job

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

# Example Slurm Job Script

jobscript.slurm

```
#!/bin/bash -l
#SBATCH --clusters=wise
#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
```

Shebang

Resource List

```
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 .
```

Module load(s)

Move data

```
python modelling.py
```

Execute commands

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

Move data

# Batch Workload

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

Job Script

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

Command Line

```
$ sbatch simulation.slurm
```

JobID

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

stderr, stdout

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

# Output File

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

## STDOUT + STDERR

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

## Out of Memory

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

## Short Walltime

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

## Low Disk Space

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

# Output File

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

stdout

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

Output File

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

Resource  
Summary

Stdout



# Output File

## Example

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

## Diagnosis

Why is my job in pending/hold state?

Check out the “Reason” on Slurm docs: [https://slurm.schedmd.com/resource\\_limits.html](https://slurm.schedmd.com/resource_limits.html)

# 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](http://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`)



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



# 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`

# 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
```

# Questions

Helpdesk:

[hpcinfo@kuleuven.be](mailto:hpcinfo@kuleuven.be) or [https://admin.kuleuven.be/icts/HPCinfo form/HPC-info-formulier](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:

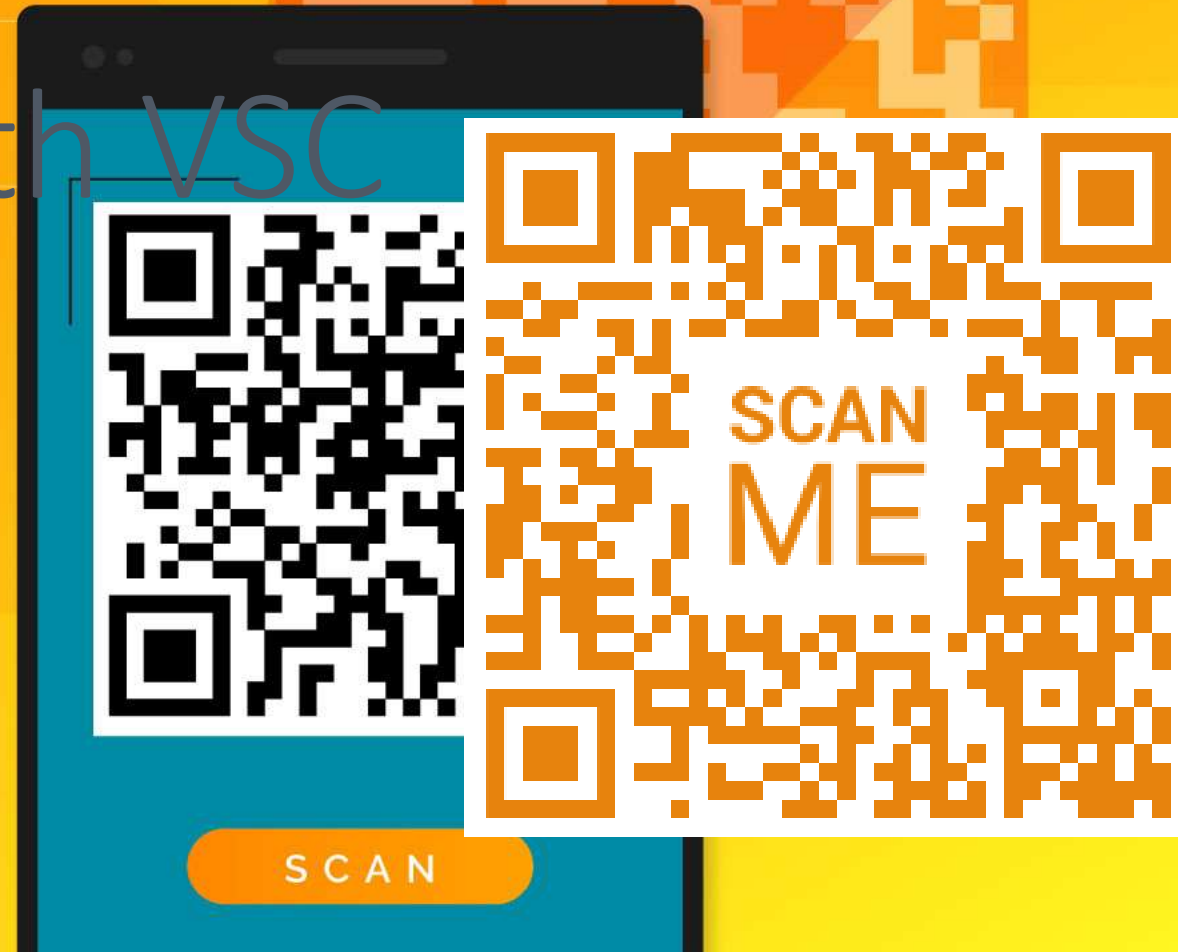
<http://status.vscentrum.be>

# Stay Connected to VSC

Start with VSC

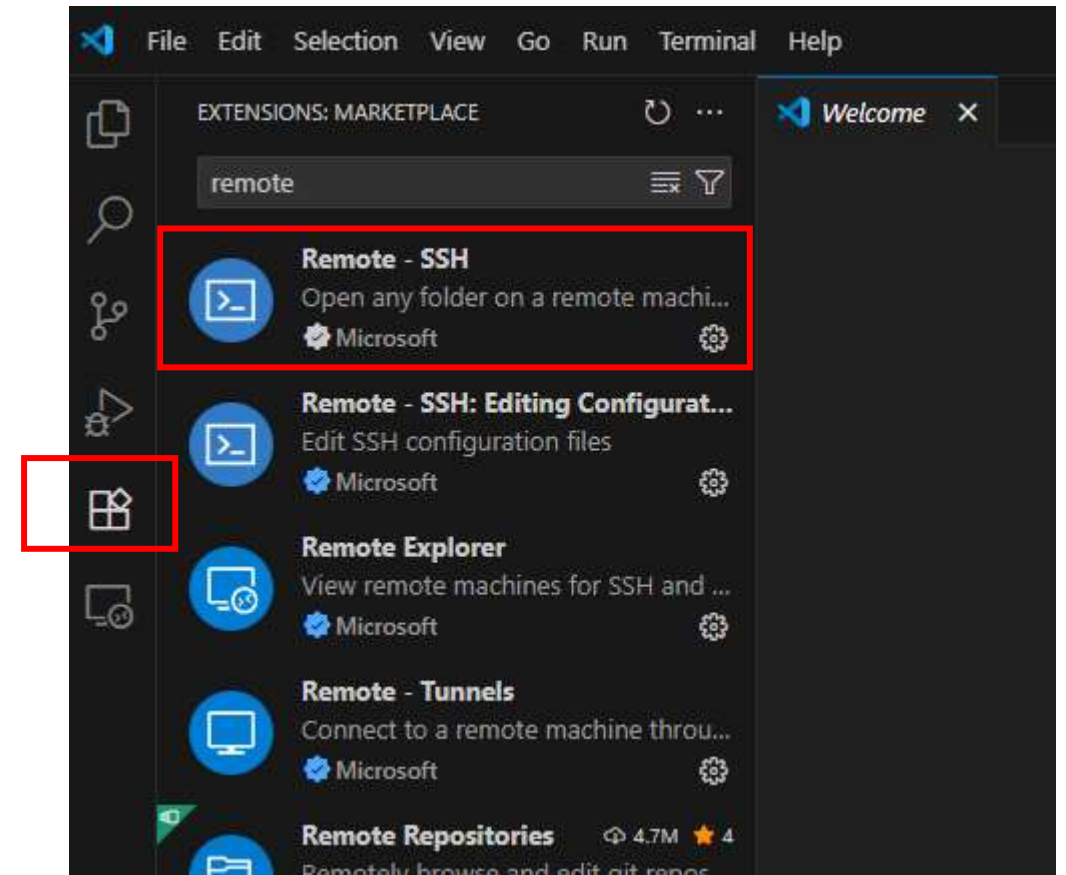


<https://hpcleuven.github.io/HPC-intro/>



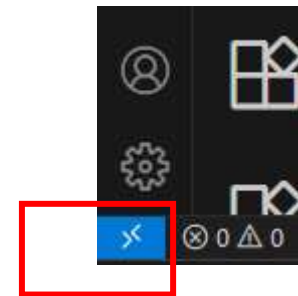
# VS Code

- ✓ Install Remote-SSH extension



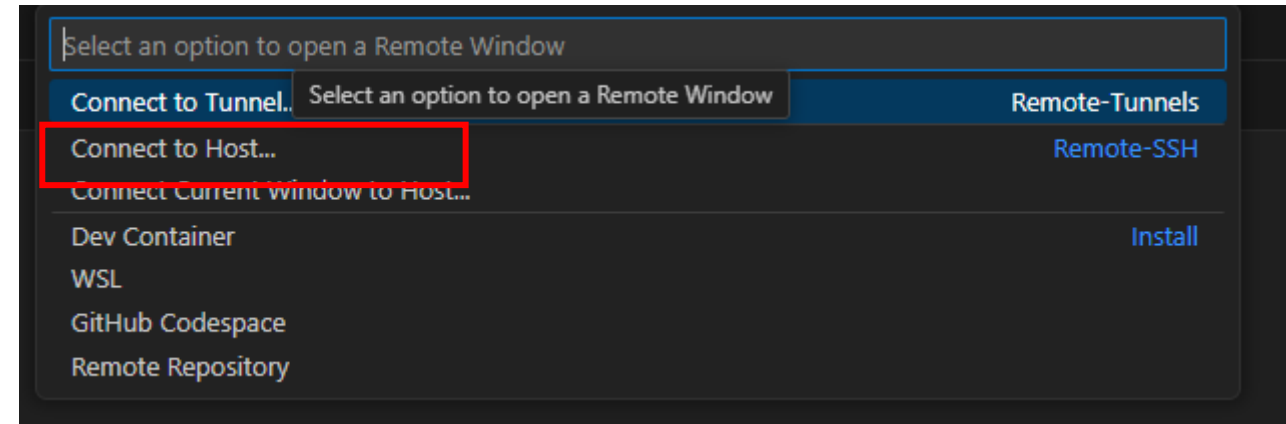
# VS Code

- ✓ Install Remote-SSH extension
- ✓ Click on the connection sign (bottom left corner)



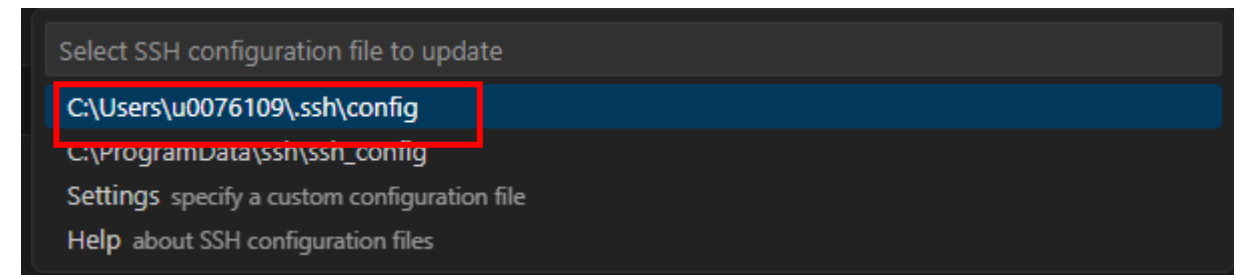
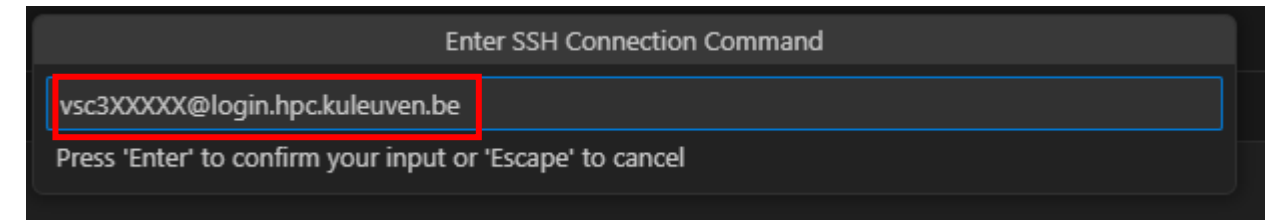
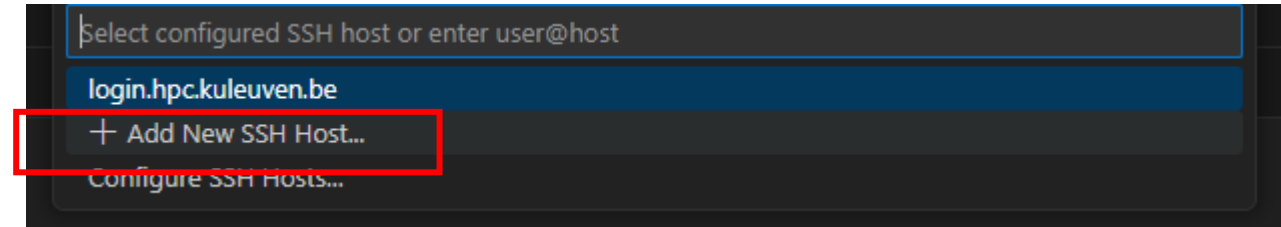
# VS Code

- ✓ Install Remote-SSH extension
- ✓ Click on the connection sign (bottom left corner)
- ✓ Click on *Connect to Host* (top of the window)



# VS Code

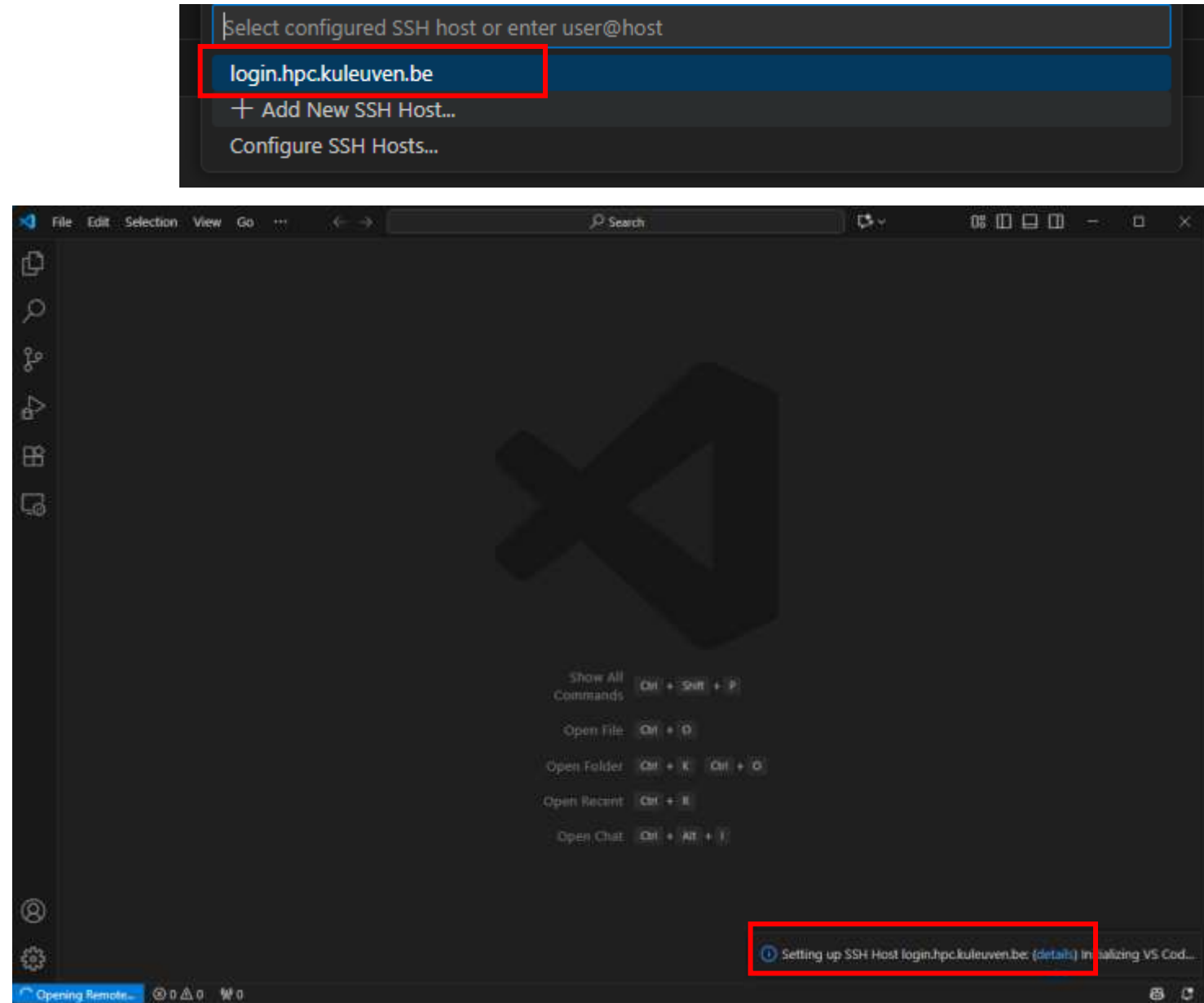
- ✓ Install Remote-SSH extension
- ✓ Click on the connection sign (bottom left corner)
- ✓ Click on *Connect to Host* (top of the window)
- ✓ Click on +Add New SSH Host
- ✓ Fill in connection details  
vsc3XXXX@login.hpc.kuleuven.be  
and press enter
- ✓ Confirm creating/updating  
configuration file
- ✓ Connection is saved





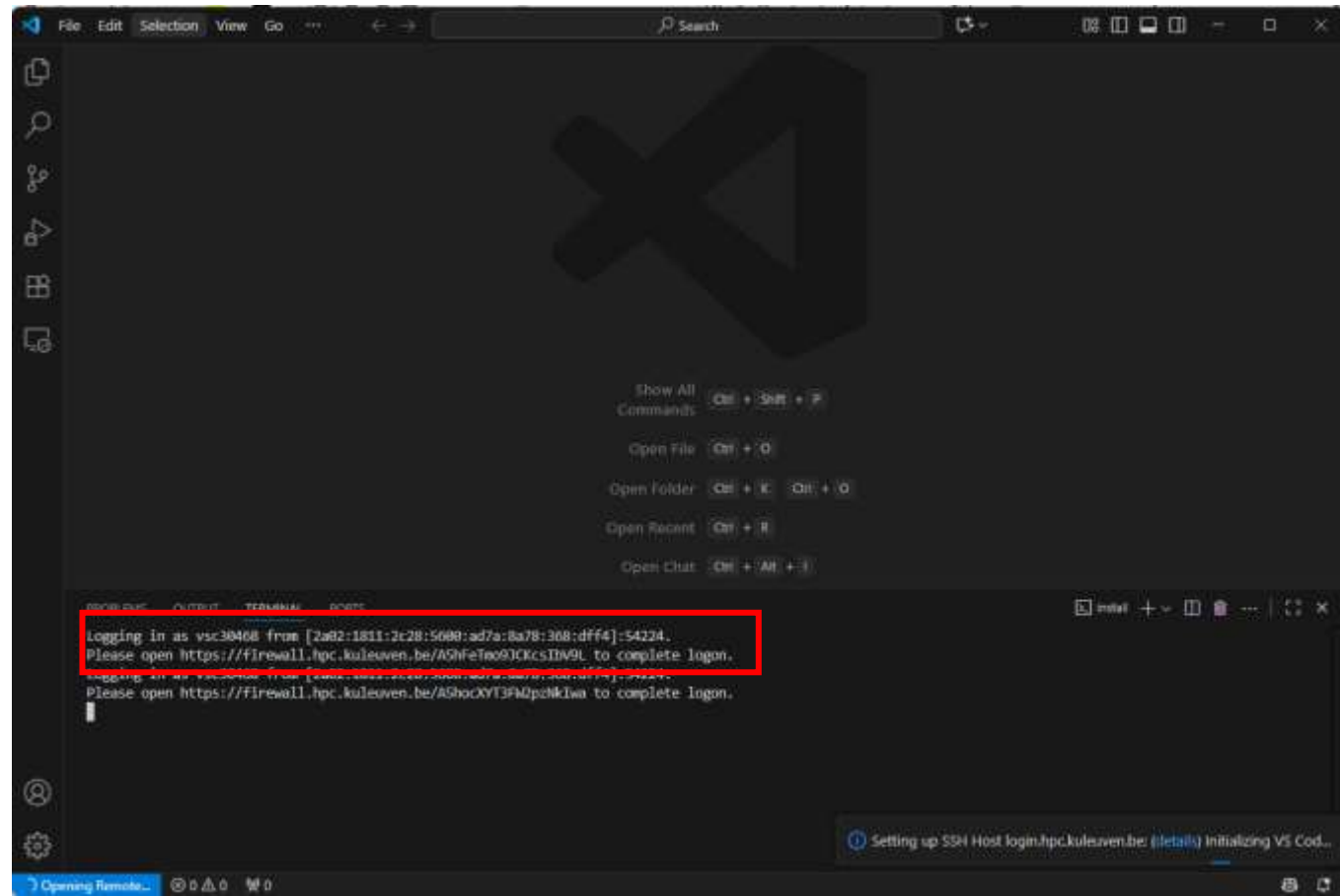
# VS Code

- ✓ Click again on the connection sign (bottom left corner)
- ✓ Select login node from the list
- ✓ New windows will open
- ✓ Click on details (in blue, bottom right corner)



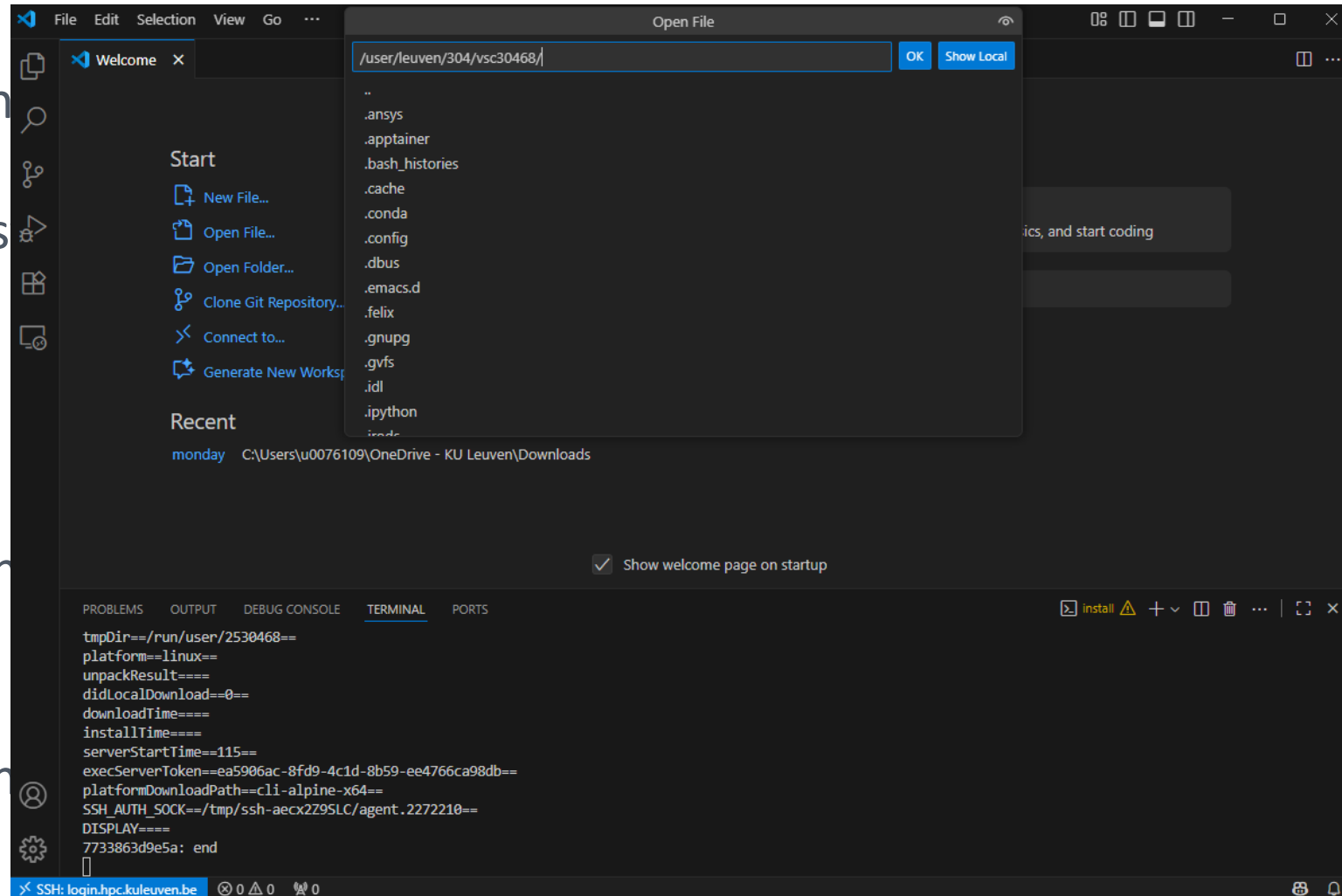
# VS Code

- ✓ Click again on the connection sign (bottom left corner)
- ✓ Select login node from the list
- ✓ New windows will open
- ✓ Click on details (in blue, bottom right corner)
- ✓ You will see firewall link to open and confirm connection



# VS Code

- ✓ Click again on the connection sign (bottom left corner)
- ✓ Select login node from the list
- ✓ New windows will open
- ✓ Click on details (in blue, bottom right corner)
- ✓ You will see firewall link to open and confirm connection
- ✓ Authenticate with MFA
- ✓ You are all set: you can create/open files on the login node of the cluster



FileEditSelectionViewGoRunTerminalHelp

EXTENSIONS

Search Extensions in Marketplace

LOCAL - INSTALLED

Remote - SSH

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...

RECOMMENDED

Debugger for Firefox

Debug your web application or bro...

Firefox DevTools

Install

WSL

Open any folder in the Windows Su...

Microsoft

Install

MCP SERVERS

MCP Servers

Disconnected from SSH: login.hpc.kuleuven.be

testsheb.pbs

Ext...

Select configured SSH host or enter user@host

login.hpc.kuleuven.be

+ Add New SSH Host...

Configure SSH Hosts...

Select configured SSH host or enter user@host

>

Microsoftmicrosoft.com29,189,049★★★★☆ (203)

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

UninstallSwitch to Pre-Release VersionAuto Update

DETAILSFEATURESEXENSION 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

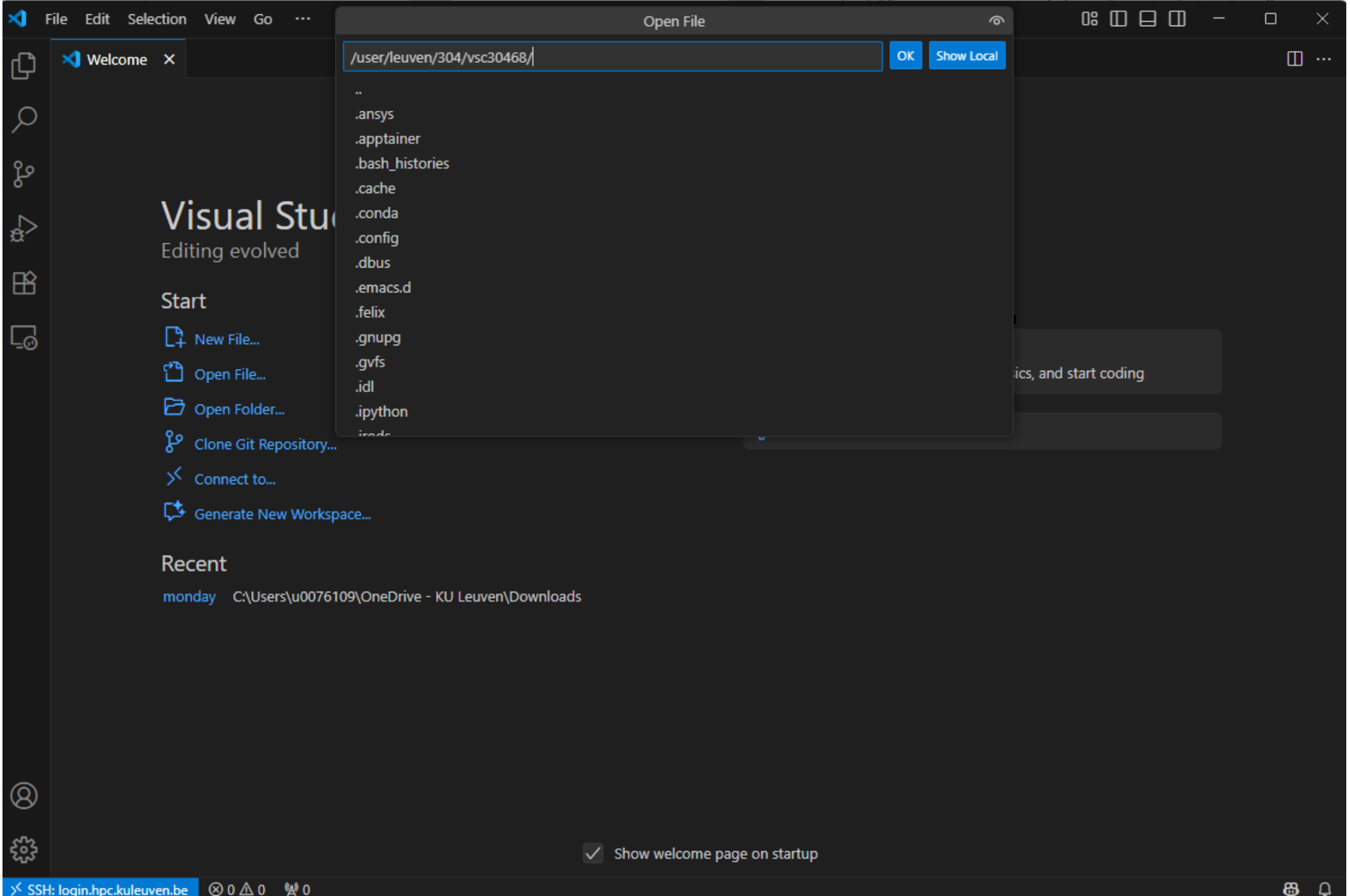
Repository

License

Microsoft

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

Install and Reload



# 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](https://people.sc.fsu.edu/~jburkardt/data/netcdf/simple_xy.nc)
- `$ ncview simple_xy.nc`

