

# Getting Started With GLiCID: Beginner Session

Mir Junaid

January 31, 2024



# Outline: Whole Session

- Introduction to High Performance Computing (HPC)
- Introducing NAUTILUS
- Working With A Supercomputer
- SSH Connections and Access to GLiCID
- SLURM Workload Manager
- Modules/Software Stack
- Guix Package Manager
- Data Management on GLiCID
- Anaconda Distribution/Micromamba
- Apptainer Containers
- Hands-on Everything

# Outline: Beginner Session

- Introduction to High Performance Computing (HPC)
  - What's HPC?
  - HPC Use Cases
- Introducing NAUTILUS
  - Architecture of Nautilus
- Working With A Supercomputer
  - Basic Linux Commands
  - SSH Connections and access to Nautilus
- SLURM Workload Manager
  - Basic Slurm Commands
  - Batch Scripting
- Modules
- Guix Package Manager
- Data Management

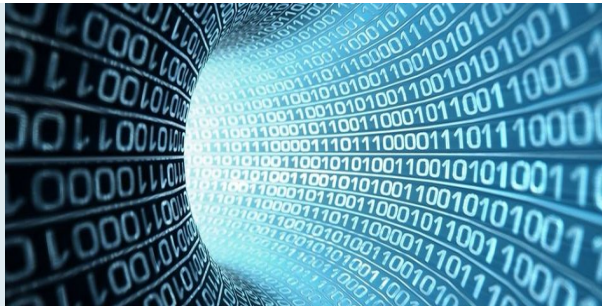
# What's HPC?

# Data, Data, Everywhere

- Key Statistics 2023
  - **3.5 quintillion bytes of data is created every single day** (Source: Earthweb)
  - 333.2 billion emails are sent per day
  - 100 billion messages are sent through WhatsApp in a day
  - 5 billion Snapchat videos and photos are shared per day
  - 456,000 tweets are made on Twitter each minute of the day
  - 500 million daily story users on Instagram every day
  - People spend \$1 million per minute online

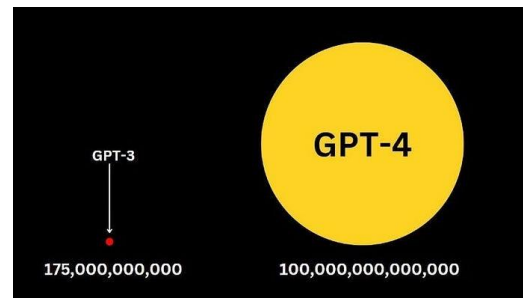


Data, data  
everywhere,  
but not a byte  
to use.



There are only 10 types  
of people in the world:  
Those who understand binary  
and those who don't.

# Data, Data, Everywhere



## Tencent ML Images



Tencent AI has now released the largest open-source, multi-label image dataset – **Tencent ML Images**. It contains nearly 18 million images, multi-labeled with up to 11,166 categories.

 Neurohive.io  
<https://neurohive.io/en/datasets/tencent-dataset/>

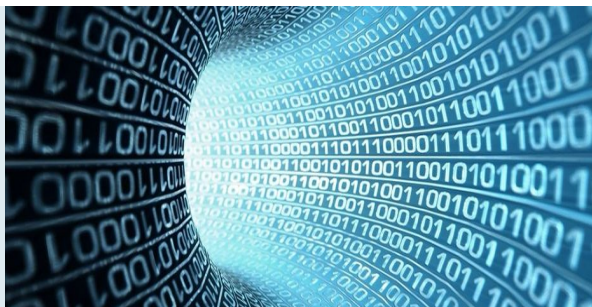
Tencent Released The Largest Multi-Labelled Image Dataset - neur...

We propose **EAGLE**, a large-scale dataset of ~1.1 million 2D meshes resulting from simulations of unsteady fluid dynamics caused by a moving flow source interacting with nonlinear scene structure, comprised of 600 different scenes of three different types.

<https://eagle-dataset.github.io/>  
**EAGLE Dataset**



Data, data  
everywhere,  
but not a byte  
to use.



There are only 10 types  
of people in the world:  
Those who understand binary  
and those who don't.

# What to do with this data?

- It is through data that
  - groundbreaking scientific discoveries are made,
  - game-changing innovations are fueled, and
  - quality of life is improved for billions of people around the globe.
- But we need huge computing power/resources to analyze this humongous data
- **HPC** gives us the power to deal with this data



# What is High Performance Computing (HPC)?

- HPC is the ability to process data and perform complex calculations at high speeds
- Laptop/desktop (3 GHz processor) can perform around 3 billion calculations/sec
- HPC solutions can perform quadrillions of calculations/sec (million times faster)
- **HPC** is the foundation for scientific, industrial, and societal advancements



# What is High Performance Computing (HPC)?

- Best-known types of HPC solutions is the **Supercomputer**
- It is made up of thousands of computers that work together
- Fastest Supercomputer is the US-based Frontier, with a processing speed of **1.102 exaflops, or quintillion floating point operations per second (flops)**
- HPC solutions can be deployed on-premise, at the edge, or even in the cloud



# Top 500

| Rank | System  | Cores     | Rmax<br>(PFlop/s) | Rpeak<br>(PFlop/s) | Power<br>(kW) |
|------|---|-----------|-------------------|--------------------|---------------|
| 1    | <b>Frontier</b> - HPE Cray EX235a, AMD Optimized 3rd Generation EPYC 64C 2GHz, AMD Instinct MI250X, Slingshot-11, HPE<br>DOE/SC/Oak Ridge National Laboratory<br>United States      | 8,699,904 | 1,194.00          | 1,679.82           | 22,703        |
| 2    | <b>Supercomputer Fugaku</b> - Supercomputer Fugaku, A64FX 48C 2.2GHz, Tofu interconnect D, Fujitsu<br>RIKEN Center for Computational Science<br>Japan                               | 7,630,848 | 442.01            | 537.21             | 29,899        |
| 3    | <b>LUMI</b> - HPE Cray EX235a, AMD Optimized 3rd Generation EPYC 64C 2GHz, AMD Instinct MI250X, Slingshot-11, HPE<br>EuroHPC/CSC<br>Finland   | 2,220,288 | 309.10            | 428.70             | 6,016         |
| 4    | <b>Leonardo</b> - BullSequana XH2000, Xeon Platinum 8358 32C 2.6GHz, NVIDIA A100 SXM4 64 GB, Quad-rail NVIDIA HDR100 Infiniband, Atos<br>EuroHPC/CINECA<br>Italy                    | 1,824,768 | 238.70            | 304.47             | 7,404         |
| 5    | <b>Summit</b> - IBM Power System AC922, IBM POWER9 22C 3.07GHz, NVIDIA Volta GV100, Dual-rail Mellanox EDR Infiniband, IBM<br>DOE/SC/Oak Ridge National Laboratory<br>United States | 2,414,592 | 148.60            | 200.79             | 10,096        |

# How does HPC work?

- A standard computing system solves problems primarily using **serial computing**
- It divides the workload into a sequence of tasks, and then executes the tasks one after the other on the same processor
- In contrast, HPC leverages
  - **Massively parallel computing**
  - **Compute clusters (also called HPC clusters)**
  - **High-performance components**

# How does HPC work?

- **Massively parallel computing**
  - Parallel computing using tens of thousands to millions of cores
- **Compute clusters/HPC clusters**
  - Consists of multiple high-speed computer servers networked together
  - The computers, called nodes, use either high-performance multi-core CPUs or, more likely today, GPUs (graphical processing units)
  - Well suited for rigorous computations and graphics-intensive tasks
- **High-performance components**
  - Other computing resources in an HPC cluster - networking, memory, storage and file systems - are **high-speed, high-throughput** and **low-latency** components that can keep pace with the nodes and optimize the computing power and performance of the cluster

# HPC: Use cases

- **AI and ML**
  - HPC supports training deep neural networks, processing large datasets, and accelerating machine learning algorithms
- **Weather and Climate Modelling**
  - HPC is used to run complex atmospheric models, simulate weather patterns, and predict climate change phenomena
- **Engineering and Design Optimization**
  - HPC is employed to optimize engineering designs, analyze structural integrity, simulate fluid dynamics, and enhance product performance
- **Astrophysics and Cosmology**
  - HPC facilitates large-scale simulations of the universe, including galaxy formation, stellar evolution, and gravitational wave analysis

# HPC: Use cases

- **Drug Discovery and Molecular Dynamics**
  - HPC enables the simulation of drug interactions, protein folding, and molecular dynamics, aiding in the development of new pharmaceuticals
- **Financial Modeling and Risk Analysis**
  - HPC helps in analyzing complex financial models, running Monte Carlo simulations, and assessing investment risks
- And many more...



# Introducing Nautilus



# Name: Why Nautilus?

- **Nautilus** is the fictional submarine belonging to [Captain Nemo](#) featured in [Jules Verne's](#) novels [Twenty Thousand Leagues Under the Seas](#) (1870) and [The Mysterious Island](#) (1874).
- Verne took the name "Nautilus" from one of the [earliest successful submarines](#), built in 1800 by [Robert Fulton](#), who also invented the first commercially successful [steamboat](#).





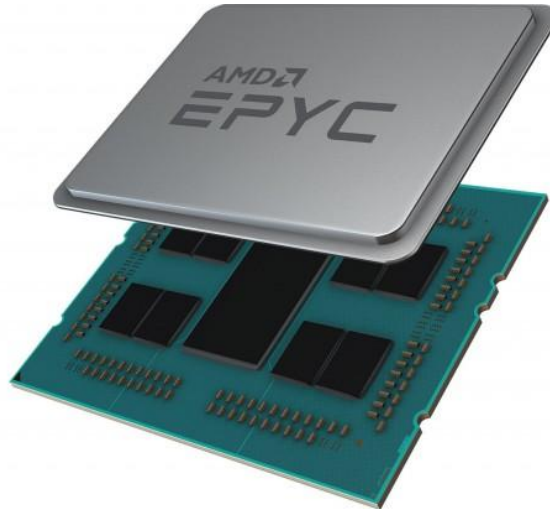
# Nautilus Architecture

- Nautilus has 3 main components:
  - Set of nodes communicating with each other
  - Fast interconnect using Infiniband 100 Gb/s technology with high bandwidth and low latency
  - Shared Storage (scratch) 427 TB (IBM/Spectrum Scale- GPFS)



# Nautilus Architecture

- Each node consists
  - Red Hat Operating System (RHEL 8.7)
  - 2 AMD EPYC 9474F processors @3.6GHz (4.1GHz Max) with 48 CPU cores
  - TDP (Thermal Design Power)/Power Consumption: 360W
  - 384 GB RAM



# Nautilus Architecture

| #Computing nodes         | Processor and Speed   | RAM    | #Cores |
|--------------------------|---|--------|--------|
| <b>40</b> cnode[301-340] | BullSequana X440 (2 AMD EPYC 9474@3.6GHz 48c)   | 384 GB | 3840   |
| <b>8</b> cnode[701-708]  | BullSequana X440 (2 AMD EPYC 9474@3.6GHz 48c)   | 768 GB | 768    |
| <b>4</b> visu[1-4]       | BullSequana X450 (2 AMD EPYC 9474@3.6GHz 48c)<br>with Nvidia A40 (48G) 2 GPUs per node    | 768 GB | 384    |
| <b>4</b> gnode[1-4]      | 4 BullSequana X410 (2 AMD EPYC 9474@3.6GHz<br>48c) with Nvidia A100 (80G) 4 GPUs per node | 768 GB | 384    |

# Philias/MesoNET

## NANTES

- Bull Sequana X440 X 30 Compute Nodes (+2)
  - Intel Sapphire Rapids (48 cores, 2.1 GHz) X 2
  - 256GB DDR (+2\*2TB DDR) + 960GB SSD
- Bull Sequana X450 Display Nodes X 2
  - Intel Sapphire Rapids (48 cores, 2.1 GHz) X 2
  - 512 GB DDR + 960 GB SSD
  - Nvidia A40 GPU 48GB x 2
- DLC Cooling
- Network: IB 100 Gb + 25 Gb eth
- GPFS: 285 TB usable
- 1 login
- 5 years of maintenance
- Available December 2023

# **Working With A Supercomputer**

# Working With A Supercomputer

Is NOT like this...



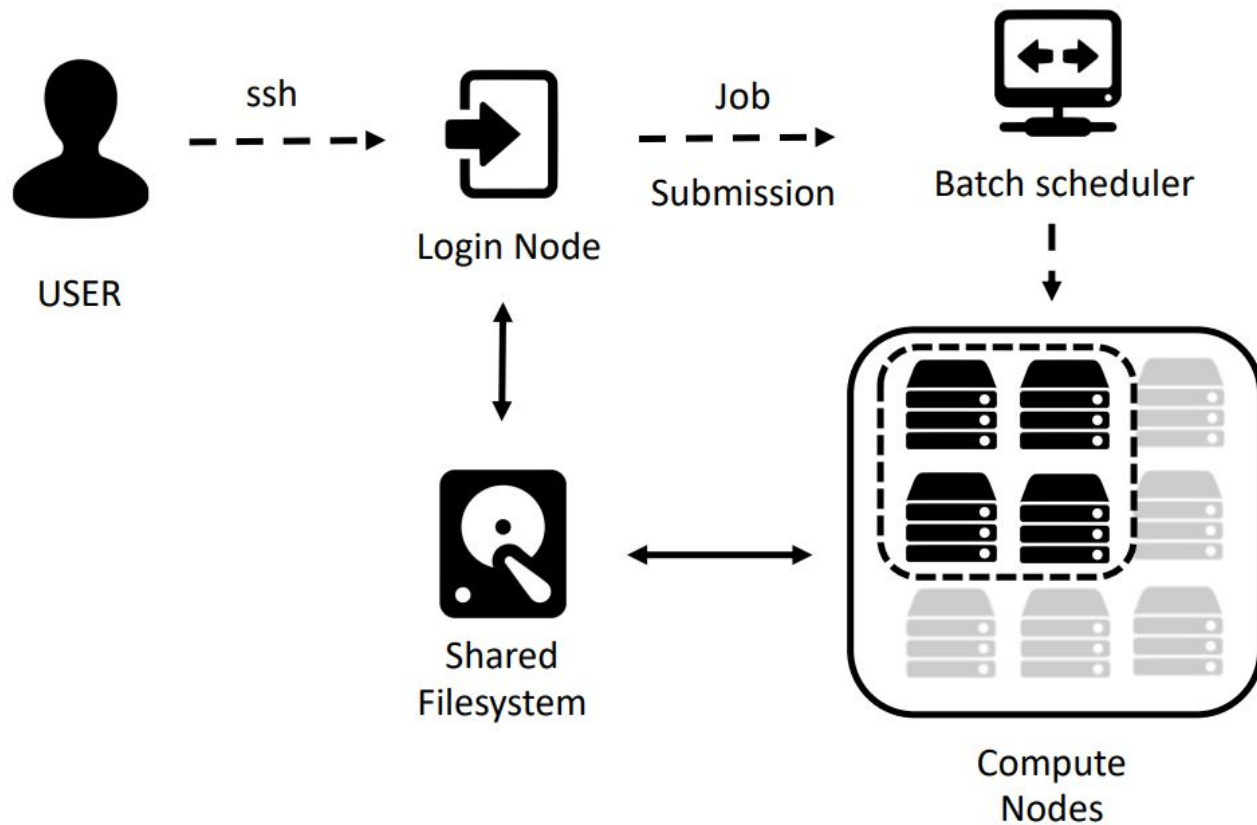
# Working With A Supercomputer







# Working With A Supercomputer



# Working With A Supercomputer



## Login node(s)

- Editing and transferring files
- Compile programs
- Prepare simulations



## Compute nodes

- Multicore nodes
- Large memories
- High-speed interconnections



## Batch scheduler

- Resource allocation
- Job queueing
- Accounting and



## File system

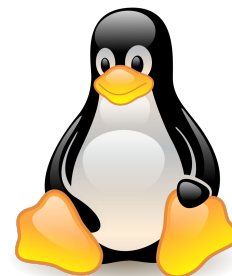
- Parallel FS
- Efficient I/O
- Node local disks

# Getting Started with GLiCID

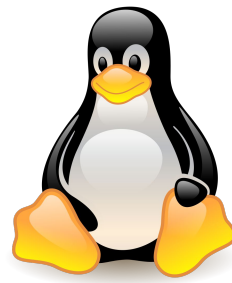
# Prerequisites

- macOS
  - Terminal (pre-installed)
- Windows
  - MobaXterm
  - PowerShell
- Linux
  - You are already well equipped :)

# LINUX COMMAND LINE



# Linux Command Line - Brief History



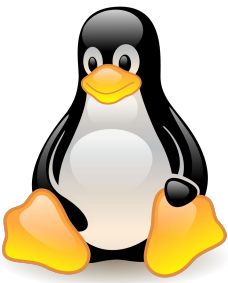
- One of the earliest operating systems was called **Unix**
- Designed to run as a multi-user system on **mainframe computers**
- Users connecting to it remotely via individual terminals
- Terminals were pretty basic: just a keyboard and screen
- Send keystrokes to the server and display any data they received on the screen
- **No mouse, no fancy graphics, not even any choice of colour**
- **Everything was sent as text, and received as text**
- Programs that ran on the mainframe had to produce text as an output and accept text as an input

# Linux Command Line - Brief History



IBM Mainframe, Late 1960's/Early 1970's

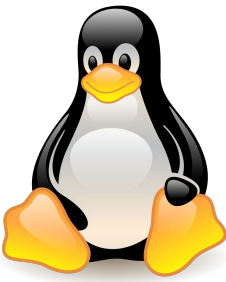
# Linux Command Line - Brief History



- **Linux** is a sort-of-descendant of Unix
- The core part of Linux is designed to behave similarly to a Unix system
- Most of the old shells and other text-based programs run on it quite happily
- Most of the [Top 500](#) supercomputers use Linux



# What's A Command Line?

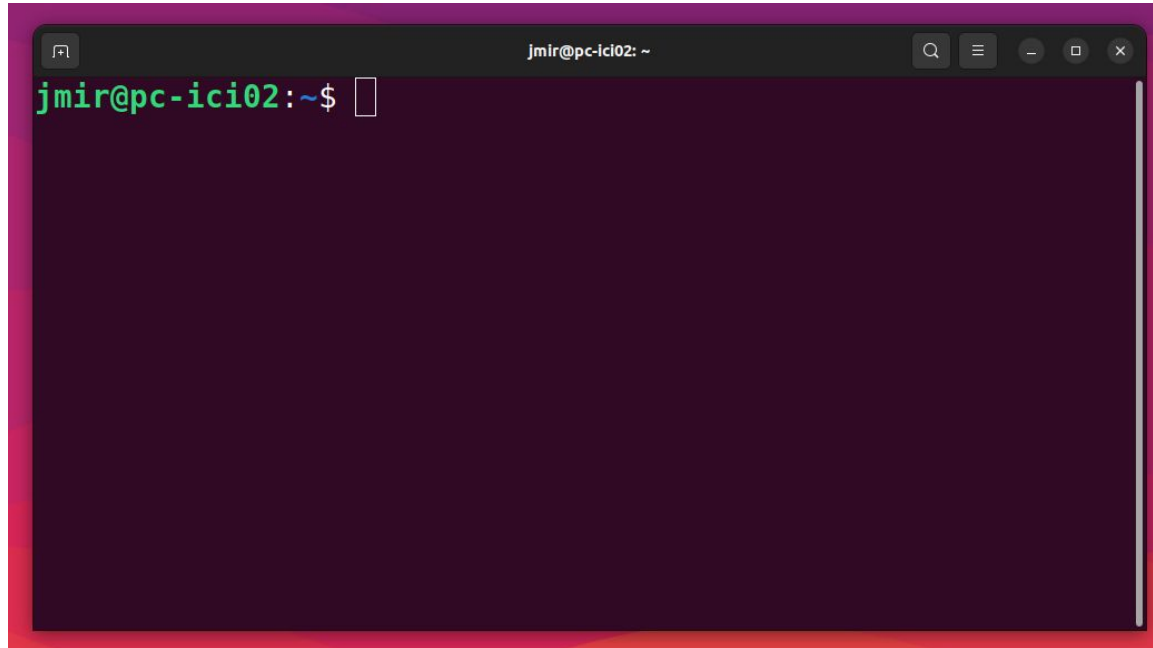


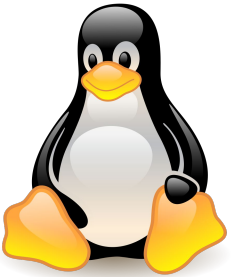
- The Linux command line is a text interface to your computer
- Often referred to as **shell, terminal, console, prompt** or various other names
- It can give the appearance of being complex and confusing to use
- But it is not so scary as it looks
- You just need to memorize a few basic commands



# Basic Linux Commands

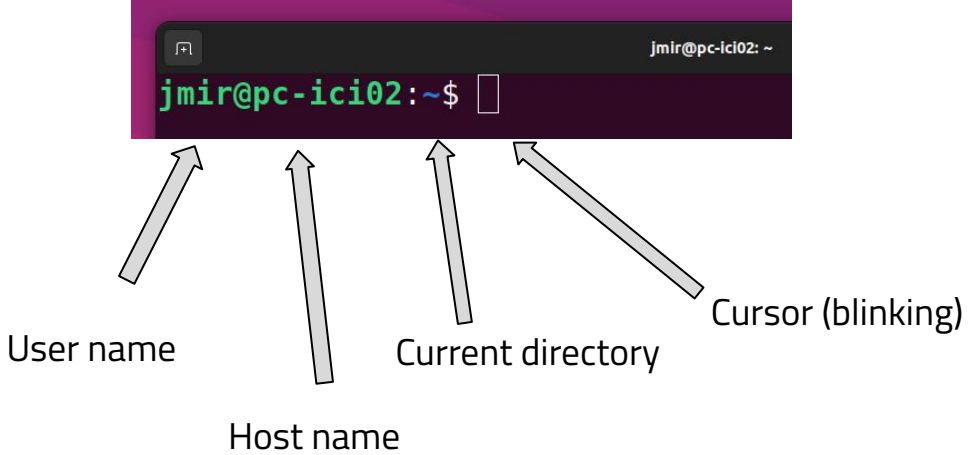
- Launch the Terminal



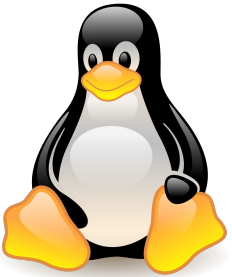


# Basic Linux Commands

- Structure of a linux commands



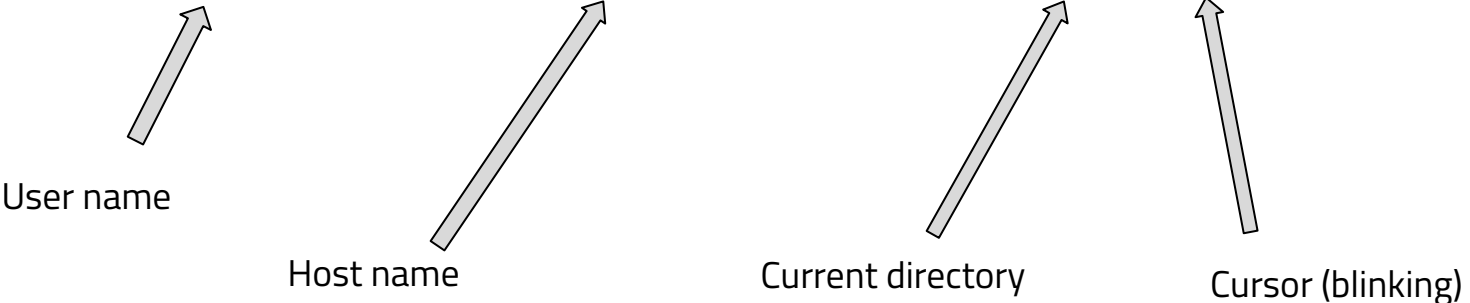
- The system is ready to accept commands



# Basic Linux Commands

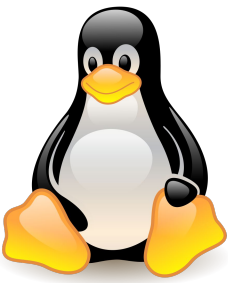
- Structure of a linux commands (in Nautilus)

```
[jmir@ec-nantes.fr@nautilus-devel-001 ~]$ █
```



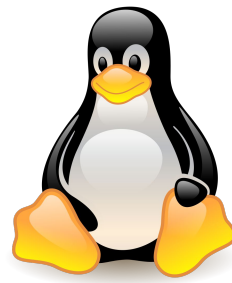
- The system is ready to accept commands

# Basic Linux Commands



| Command                 | Syntax             | Description  |
|-------------------------|--------------------|--|
| Print Working Directory | <code>pwd</code>   | Print present working directory                      |
| List                    | <code>ls</code>    | List files and directories at path                   |
| Change directory        | <code>cd</code>    | Change current directory                             |
| Make directory          | <code>mkdir</code> | Create new directory                                 |
| Create empty file       | <code>touch</code> | Create new file or update timestamp                  |
| Move                    | <code>mv</code>    | Move or rename files and directories                 |
| Copy                    | <code>cp</code>    | Copy files or directories from source to destination |
| Remove                  | <code>rm</code>    | Remove files   |
| Text editor             | <code>vim</code>   | Vim is a highly configurable text editor             |

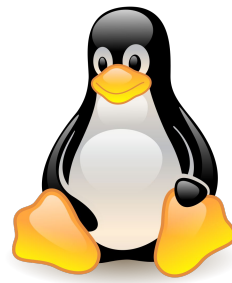
# Basic Linux Commands



| Command                 | Syntax           | Description                     |
|-------------------------|------------------|---------------------------------|
| Print Working Directory | <code>pwd</code> | Print present working directory |

```
jmir@pc-ici02: ~  
jmir@pc-ici02:~$ pwd  
/home/jmir  
jmir@pc-ici02:~$
```

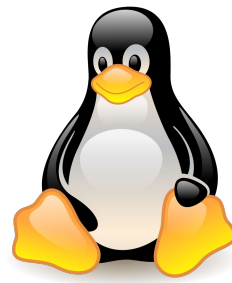
# Basic Linux Commands



| Command | Syntax          | Description                        |
|---------|-----------------|------------------------------------|
| List    | <code>ls</code> | List files and directories at path |

```
jmir@pc-ici02: ~$ ls
access-token.txt  git_repo          Templates
anaconda3         inventory.yaml    teskey.txt
cv_debug.log      Music             test2.txt
Desktop           nautilus-tutorial testdir
Documents         Pictures           testscript.sh
Downloads         playbook.yaml     test.txt
ghh               Public            ust4hpc
ghh.pub          snap              Videos
jmir@pc-ici02: ~$
```

# Basic Linux Commands

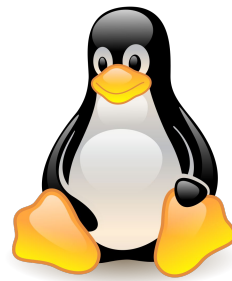


| Command          | Syntax          | Description              |
|------------------|-----------------|--------------------------|
| Change directory | <code>cd</code> | Change current directory |

```
jmir@pc-ici02: ~/nautilus-tutorial
jmir@pc-ici02:~$ cd nautilus-tutorial/
jmir@pc-ici02:~/nautilus-tutorial$
```



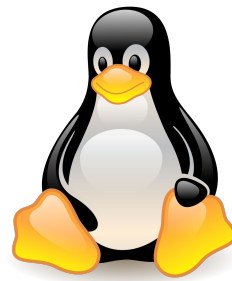
# Basic Linux Commands



| Command        | Syntax             | Description          |
|----------------|--------------------|----------------------|
| Make directory | <code>mkdir</code> | Create new directory |

```
jmir@pc-ici02: ~/nautilus-tutorial
jmir@pc-ici02:~/nautilus-tutorial$ ls
cheat-sheet
jmir@pc-ici02:~/nautilus-tutorial$ mkdir test-dir
jmir@pc-ici02:~/nautilus-tutorial$ ls
cheat-sheet  test-dir
jmir@pc-ici02:~/nautilus-tutorial$
```

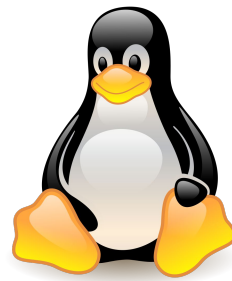
# Basic Linux Commands



| Command           | Syntax             | Description                         |
|-------------------|--------------------|-------------------------------------|
| Create empty file | <code>touch</code> | Create new file or update timestamp |

```
jmir@pc-ici02: ~/nautilus-tutorial
jmir@pc-ici02:~/nautilus-tutorial$ touch test-file
jmir@pc-ici02:~/nautilus-tutorial$ ls
cheat-sheet test-dir test-file
jmir@pc-ici02:~/nautilus-tutorial$
```

# Basic Linux Commands



| Command | Syntax          | Description                          |
|---------|-----------------|--------------------------------------|
| Move    | <code>mv</code> | Move or rename files and directories |

```
jmir@pc-ici02: ~/nautilus-tutorial/cheat-sheet
jmir@pc-ici02:~/nautilus-tutorial$ ls
cheat-sheet  test-dir
jmir@pc-ici02:~/nautilus-tutorial$ mv test-dir/ cheat-sheet/
jmir@pc-ici02:~/nautilus-tutorial$ ls
cheat-sheet
jmir@pc-ici02:~/nautilus-tutorial$ cd cheat-sheet/
jmir@pc-ici02:~/nautilus-tutorial/cheat-sheet$ ls
test-dir  test-file
jmir@pc-ici02:~/nautilus-tutorial/cheat-sheet$
```

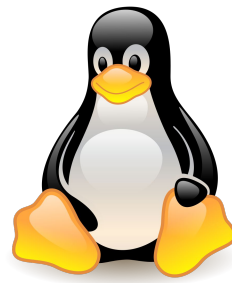
# Basic Linux Commands



| Command | Syntax          | Description  |
|---------|-----------------|--|
| copy    | <code>cp</code> | Copy files or directories from source to destination |

```
Jmir@pc-ici02: ~/nautilus-tutorial/cheat-sheet
jmir@pc-ici02:~/nautilus-tutorial$ ls
cheat-sheet  test-file
jmir@pc-ici02:~/nautilus-tutorial$ cp test-file cheat-sheet/
jmir@pc-ici02:~/nautilus-tutorial$ cd cheat-sheet/
jmir@pc-ici02:~/nautilus-tutorial/cheat-sheet$ ls
test-dir  test-file
jmir@pc-ici02:~/nautilus-tutorial/cheat-sheet$
```

# Basic Linux Commands

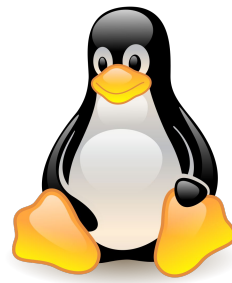


| Command | Syntax          | Description  |
|---------|-----------------|--------------|
| Remove  | <code>rm</code> | Remove files |

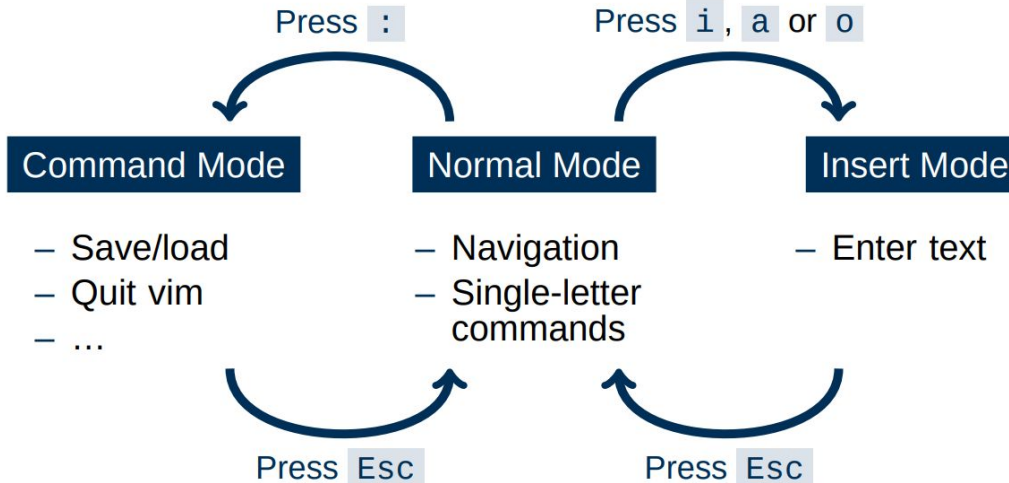
```
jmir@pc-ici02: ~/nautilus-tutorial/cheat-sheet
jmir@pc-ici02:~/nautilus-tutorial/cheat-sheet$ ls
test-dir  test-file
jmir@pc-ici02:~/nautilus-tutorial/cheat-sheet$ rm test-file
jmir@pc-ici02:~/nautilus-tutorial/cheat-sheet$ ls
test-dir
jmir@pc-ici02:~/nautilus-tutorial/cheat-sheet$
```



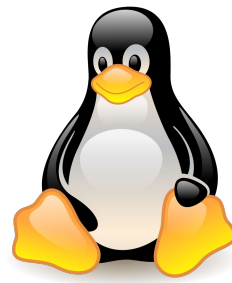
# Basic Linux Commands



| Command      | Syntax           | Description                              |
|--------------|------------------|--|
| Linux editor | <code>vim</code> | Vim is a highly configurable text editor |



# Basic Linux Commands



| Command     | Syntax | Description                        |
|-------------|--------|------------------------------------|
| Text editor | nano   | Comparatively easier (Ctrl+Option) |

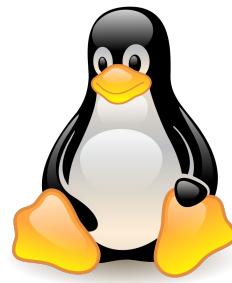
```
jmlr@pc-ic102: ~/nautilus-tutorial
GNU nano 6.2 test_script.sh
#!/bin/bash
mkdir new-dir && cd new-dir
echo "Ciao"
```

[ Read 4 lines ]

^G Help    ^O Write Out    ^W Where Is    ^K Cut    ^T Execute    ^C Location    M-U Undo  
^X Exit    ^R Read File    ^\ Replace    ^U Paste    ^J Justify    ^\_ Go To Line    M-E Redo



# Basic Linux Commands



| Command     | Syntax           | Description                          |
|-------------|------------------|--------------------------------------|
| User Manual | <code>man</code> | Displays whole manual of the command |

```
jmir@pc-ici02: ~/nautilus-tutorial
jmir@pc-ici02:~/nautilus-tutorial$ man ls

LS(1)                                User Commands                                LS(1)

NAME
  ls - list directory contents

SYNOPSIS
  ls [OPTION]... [FILE]...

DESCRIPTION
  List information about the FILES (the current directory by default).
  Sort entries alphabetically if none of -cftuvSUX nor --sort is speci-
  fied.

  Mandatory arguments to long options are mandatory for short options too.

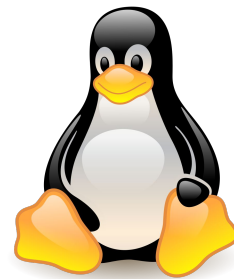
  -a, --all
      do not ignore entries starting with .

  -A, --almost-all
      do not list implied . and ..

  --author
      with -l, print the author of each file

Manual page ls(1) line 1 (press h for help or q to quit)
```

# Basic Linux Commands - File Management



Path to folders and files

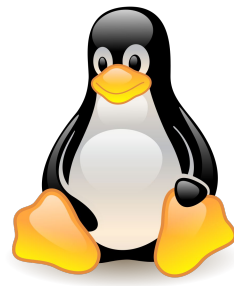
- Relative Path

```
jmir@pc-ici02: ~/nautilus-tutorial/test-dir
jmir@pc-ici02:~/nautilus-tutorial$ cd test-dir/
jmir@pc-ici02:~/nautilus-tutorial/test-dir$
```

- Absolute path

```
jmir@pc-ici02: ~/nautilus-tutorial/test-dir
jmir@pc-ici02:~/nautilus-tutorial$ cd ~/nautilus-tutorial/test-dir/
jmir@pc-ici02:~/nautilus-tutorial/test-dir$
```

# Basic Linux Commands - Large Files



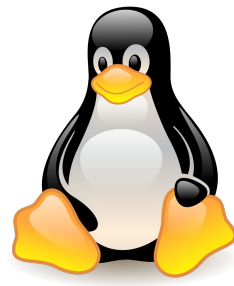
- Compress

```
$ tar -czvf <folder_name.tar.gz> <foldername>
```

- Decompress

```
$ tar -xzvf <folder_name.tar.gz>
```

# Basic Linux Commands - File Management



- Local to Remote

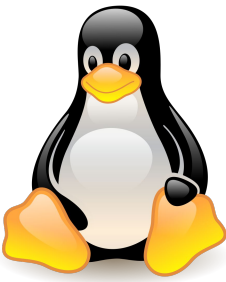
```
$ scp -r folder_name nautilus:/scratch/nautilus/users/username
```

- Remote to Local

```
$ scp -r nautilus:/scratch/nautilus/users/username/folder_name /local_location
```

Note: Run both commands from the local machine.

# Basic Linux Commands - Bash Scripting



- What if we want to run many bash commands?
- ... maybe in a workflow???
- Important part of process automation in Linux
- Plain text file that contains a series of commands
- Any command you run on the command line can be put in a script and vice-versa
- Executed like a program

# Basic Linux Commands - Bash Scripting



- Simple bash script

```
#!/bin/bash  
mkdir test-dir && cd test-dir  
echo "Ciao"
```

- Save as `test_script.sh`
- To execute `./test_script.sh`

# File Permissions



`chmod u+x <filename>`

```
jmir@pc-ici02: ~/nautilus-tutorial
jmir@pc-ici02:~/nautilus-tutorial$ touch test_script.sh
jmir@pc-ici02:~/nautilus-tutorial$ vi test_script.sh
jmir@pc-ici02:~/nautilus-tutorial$ ./test_script.sh
bash: ./test_script.sh: Permission denied
jmir@pc-ici02:~/nautilus-tutorial$ chmod u+x test_script.sh
jmir@pc-ici02:~/nautilus-tutorial$ ./test_script.sh
Ciao
jmir@pc-ici02:~/nautilus-tutorial$ ls
cheat-sheet  new-dir  python  pytorch_geometric  test-dir  test_script.sh
jmir@pc-ici02:~/nautilus-tutorial$
```

# SSH CONNECTIONS





# What is SSH Key?



- SSH is a secure shell (terminal) connection to another computer
- You connect from your computer to the LOGIN NODE
- Security is given by public/private keys
- A connection to the supercomputer needs a
  - Key,
  - Configuration,
  - Key/IP address known to the supercomputer



**ENCRYPTED COMMUNICATION**

# How to access GLiCID cluster?

- Create an account on <https://clam.glicid.fr>
- Account validation by an administrator
- User uploads SSH key to CLAM portal (in profile's SSH Access tab)
- SSH connection configuration on local PC



# How to configure SSH connection?



- Generate SSH key and copy the public key (`id_ed25519.pub`)

```
jmir@pc-ici02:~$ ssh-keygen -t ed25519
Generating public/private ed25519 key pair.
Enter file in which to save the key (/home/jmir/.ssh/id_ed25519): y
```

```
jmir@pc-ici02:~/.ssh$ cat id_ed25519.pub
ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAID7Tm0MUiYv62VbE/uyk1Gcan9Wfu1IEsg7sBX8
R6Fjw junaid.mir@ec-nanes.fr
jmir@pc-ici02:~/.ssh$
```

# How to configure SSH connection?



- Upload this SSH key to the CLAM

The screenshot shows the GLICID Cluster Account Manager interface. The user is Junaid Mir (Junaid.Mir@ec-nantes.fr). The account is activated. The interface shows the 'SSH Access' tab with a section for 'SSH-Keys'. A text area contains the following SSH key:

```
ssh-rsa
AAAAB3NzaC1yc2EAAAADAQABAAQGC3Dp8oVQrXQ9xnY6QZE01Vu26+Wv2ZbdLOkMIM8qsGyG
snjwNY62l9jU4j3xcAN5ZISVL12GtZpmstx4hzroky1fDKRRWsgLix1Du4LtnBi00+iDDghJb1nBvDYAL
w6zpbssa3KiwdogHdzWjV5ailP5zrg9iXtW3m4xOC/CzLUwqeVCMosk9IG2+jf6k1BkmBPKhwtWaAli/
v63oNde/QuWaqLpxQsihk15R0r1mOUlwW8lke5/OblYxyGqOx49FzV+2YLHVqH2uzZl1MXOpZz
KIAT7HzDKdeE6vrFyuc3d/ud7bk+j6eKbq6aWwAdO5T7J3ZA/0tS07+G8nhJr2szn1EL7AdQXR/y8
+SPQWBoXdq1qmqz2XcLY50m6649Xt3FtdShiqlmeVCLYqyDhVCol65k1C9xmiad0kOp355e6254v
KwAdFiv4UOEJXw4/jyEUJ5xWTeGbXbotCDW+LgZTERA4s+uNAFdzGFz1RTH2OqoG2jGLs9PbyGp9E=
jmir@pc-ici02
ssh-rsa
```

Below the key is an 'Upload' button. The 'SSH Configuration' section provides instructions on how to use the key to login to GLICID. It includes a terminal snippet:

```
1 Host Bastion
2 Hostname bastion.glicid.fr
3 User jmir@ec-nantes.fr
4 IdentityFile ~/.ssh/SSH_KEY_GLICID
```

At the bottom left, the version information is: CLAM - V1.0.0 CHANGELOG.

# How to configure SSH connection?



- Copy Contents to the **config file** and save it
- Replace **<my\_username>** with your username

```
jmir@pc-ici02:~/.ssh$ cat config
Host Bastion
    Hostname bastion.glicid.fr
    User jmir@ec-nantes.fr
    IdentityFile ~/.ssh/id_ed25519
    ForwardAgent yes

Host glicid
    Hostname login-001.glicid.fr
    User jmir@ec-nantes.fr
    ProxyJump Bastion
    IdentityFile ~/.ssh/id_ed25519

Host nautilus
    Hostname nautilus-devel-001.nautilus.intra.glicid.fr
    User jmir@ec-nantes.fr
    ProxyJump glicid
    IdentityFile ~/.ssh/id_ed25519
```

# SSH to GLiCID Cluster



- Login using SSH by typing this command in the terminal
  - `ssh glicid` and then press `<Enter>`

```
jmir@pc-ici02:~/.ssh$ ssh glicid
The authenticity of host 'bastion.glicid.fr (194.167.60.10)' can't be established.
ED25519 key fingerprint is SHA256:0Szy+0r30Rkizt8TXqKeLqD4qRn8Xq+0YmEE4EsfqrU.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'bastion.glicid.fr' (ED25519) to the list of known hosts.
The authenticity of host 'login-001.glicid.fr (<no hostip for proxy command>)' can't be established.
ED25519 key fingerprint is SHA256:0Szy+0r30Rkizt8TXqKeLqD4qRn8Xq+0YmEE4EsfqrU.
This host key is known by the following other names/addresses:
  ~/.ssh/known_hosts:1: [hashed name]
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'login-001.glicid.fr' (ED25519) to the list of known hosts.
Last login: Wed Nov 29 14:07:00 2023 from 194.167.60.12
jmir@ec-nantes.fr@guix-devel-001 ~$ ls
env.yml KEYS ml-container test-containers TP_ContainerWorkshop wget-log wget-log.1
jmir@ec-nantes.fr@guix-devel-001 ~$
```



# SSH to GLiCID Cluster



- Switch from glicid to nautilus
  - `ssh nautilus-devel-001` and then press `<Enter>`
  - It will not work but ask for a password

```
jmir@ec-nantes.fr@guix-devel-001 ~$ ssh nautilus-devel-001
#####
#   This service is restricted to authorized users only. All   #
#   activities on this system are logged.                       #
#   Unauthorized access will be fully investigated and reported #
#   to the appropriate law enforcement agencies.               #
#####
jmir@ec-nantes.fr@nautilus-devel-001's password: █
```



# SSH to GLiCID Cluster



- Switch from glicid to nautilus
  - To switch to nautilus, set-up `authorized_keys`
  - `cd ~/.ssh` after logging in to glicid
  - Either generate a new key or copy the private key `id_ed2259` (which is not a good idea)

```
jmir@ec-nantes.fr@guix-devel-001 ~/.ssh$ ssh-keygen -t ed25519
Generating public/private ed25519 key pair.
Enter file in which to save the key (/home/jmir@ec-nantes.fr/.ssh/id_ed25519):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/jmir@ec-nantes.fr/.ssh/id_ed25519
Your public key has been saved in /home/jmir@ec-nantes.fr/.ssh/id_ed25519.pub
The key fingerprint is:
SHA256:7bx0Ee+80Lh9ykh3/5DGxRPzPXa5DkJqQodBSU6/L0 jmir@ec-nantes.fr@guix-devel-001.waves.intra.glicid.f
r
The key's randomart image is:
+--[ED25519 256]--+
|
|   ooo  oo|
|  . .  .|=|
| + . .  0 .|=|
| +.oo . *+|
| .Sooo. o+o|
| .o.B.  o.|
| .XE+ . o |
| + 0 o.o |
| . = *+o.o|
+----[SHA256]-----+
jmir@ec-nantes.fr@guix-devel-001 ~/.ssh$ ls
id_ed25519 id_ed25519.pub known_hosts known_hosts.old
jmir@ec-nantes.fr@guix-devel-001 ~/.ssh$ cat id_ed25519.pub > authorized_keys
jmir@ec-nantes.fr@guix-devel-001 ~/.ssh$ ls
authorized_keys id_ed25519 id_ed25519.pub known_hosts known_hosts.old
jmir@ec-nantes.fr@guix-devel-001 ~/.ssh$ ssh nautilus-devel-001
```







# SSH configuration on Windows

- Open Windows PowerShell
  - Run `ssh-keygen` (Preferably use `id_ed25519`)
  - Save it in `C:\Users\username\.ssh\id_rsa` (normally it will be automatic)
  - Don't enter any passwords - just press enter (easier)
  - You'll find two files
    - `id_rsa.pub` and `id_rsa.ppk`
  - Create an account on <https://glicid.clam.fr> and upload the public key(`id_rsa.pub`) (remove spaces if any)(be careful, don't delete anything by mistake)



# SSH configuration on Windows

- Open MobaXterm
- On SSH, configure
  - Remote-host: `login-001.glicid.fr`
  - Username: `jmir@ec-nantes.fr` (DON'T USE MINE)
  - Port: 22 (automatic)
  - Click **Advanced SSH settings**
  - Use private key (upload your private key)(`id_rsa.pkk`)
  - Go to **Network Settings** -> **SSH gateway** (jump host)
    - Gateway host: `bastion.glicid.fr`
    - Username: `jmir@ec-nantes.fr` (DON'T USE MINE)
    - Port:22
    - Use SSH key -> upload private key (`id_rsa.pkk`)
    - **Click OKAY**
  - **Click OKAY** in Session Settings
  - Double click OR right click and execute a session
  - You'll be logged in.
  - To exit -> Either write exit in the terminal and press enter or click on the exit option.
- Be careful: Everything is case-sensitive and don't use MobaXterm for key generation



# SSH configuration on Windows

- Open PowerShell

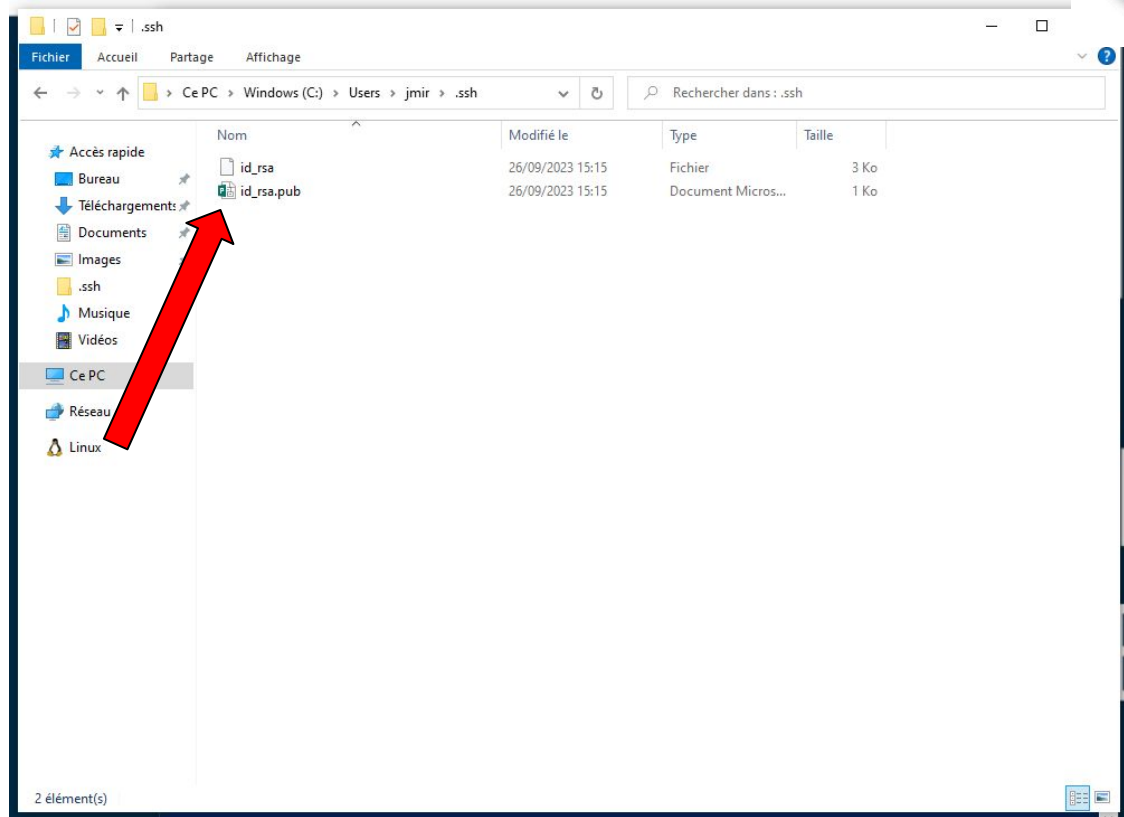
ssh-keygen

```
Windows PowerShell
PS C:\Users\jmir> ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (C:\Users\jmir\.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in C:\Users\jmir\.ssh/id_rsa.
Your public key has been saved in C:\Users\jmir\.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:n1QjWA5dcWpn0jhfrd1r1WA/7x43iuEx/S3/4WdsDE jmir@pc-gem09
The key's randomart image is:
+---[RSA 3072]-----+
|
|  .+o..
| + o .
| = o +
| oSo..+Eo .
| oo+o.+BBo
| .o.o*.B
| .ooo+
| .o++B
+----[SHA256]-----+
PS C:\Users\jmir>
```



# SSH configuration on Windows

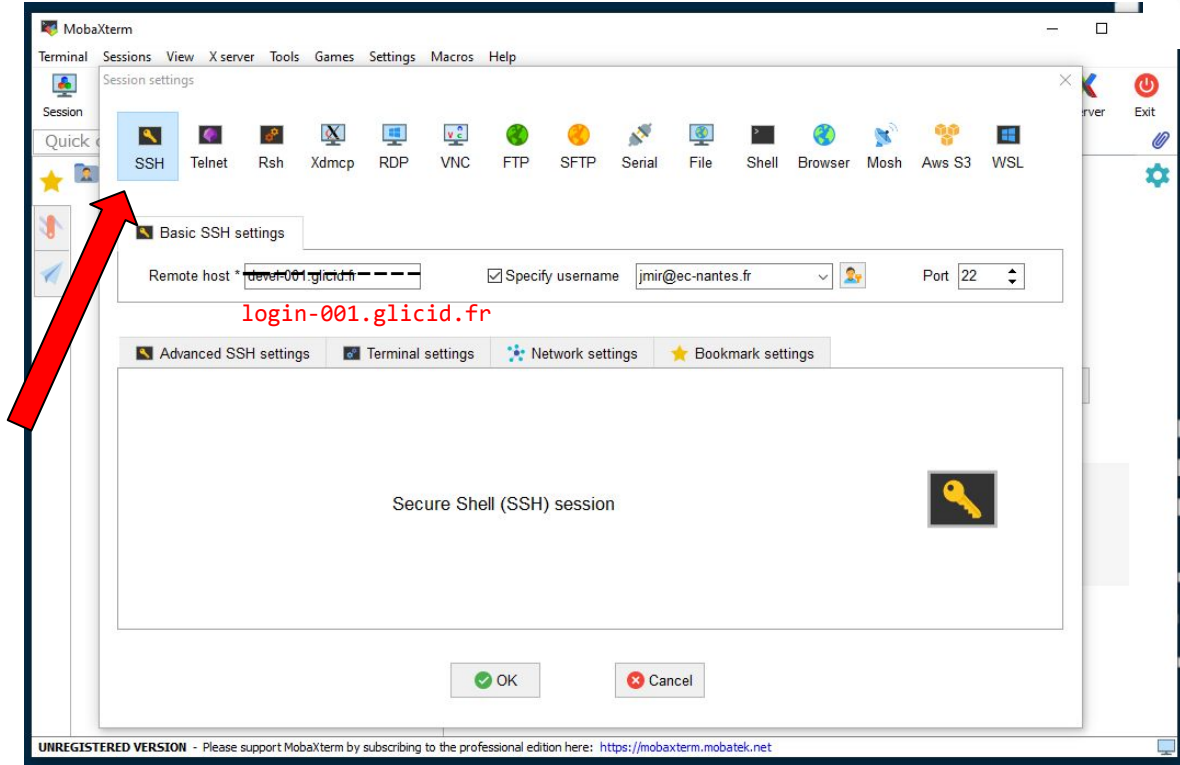
- Keys generated
  - Private key
  - Public key





# SSH configuration on Windows

- Open a session
- Configure SSH

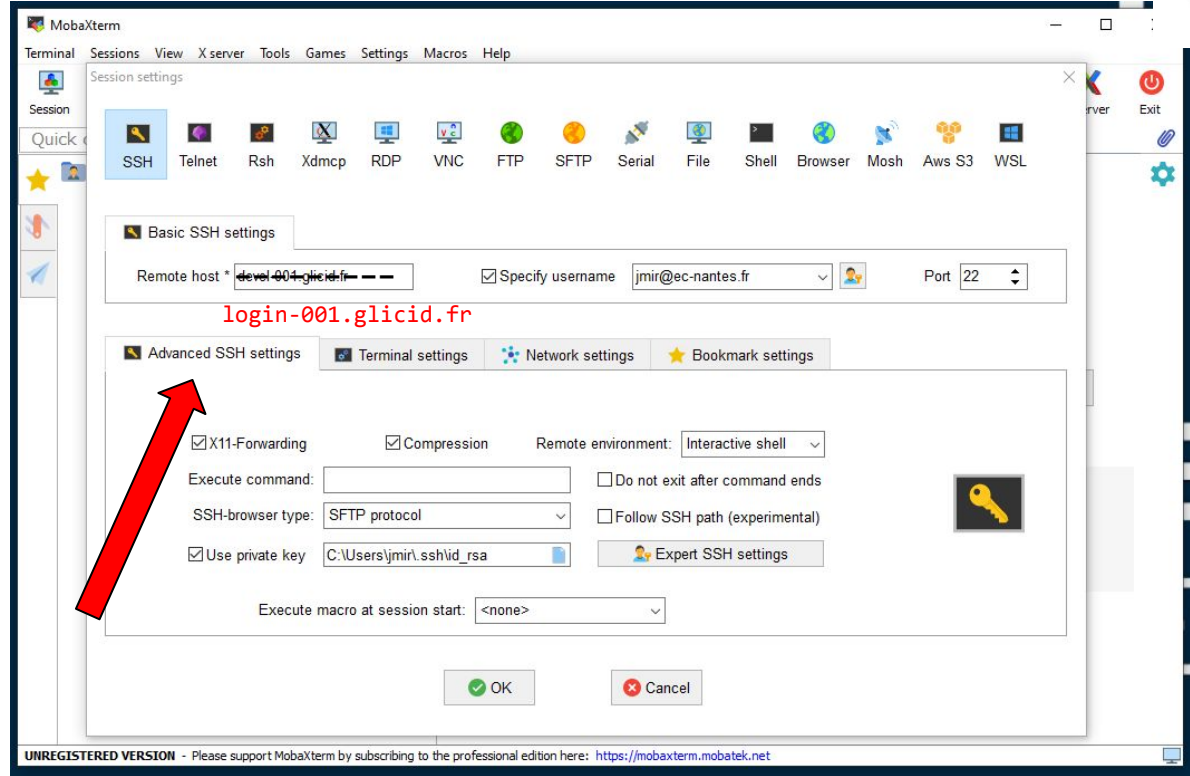






# SSH configuration on Windows

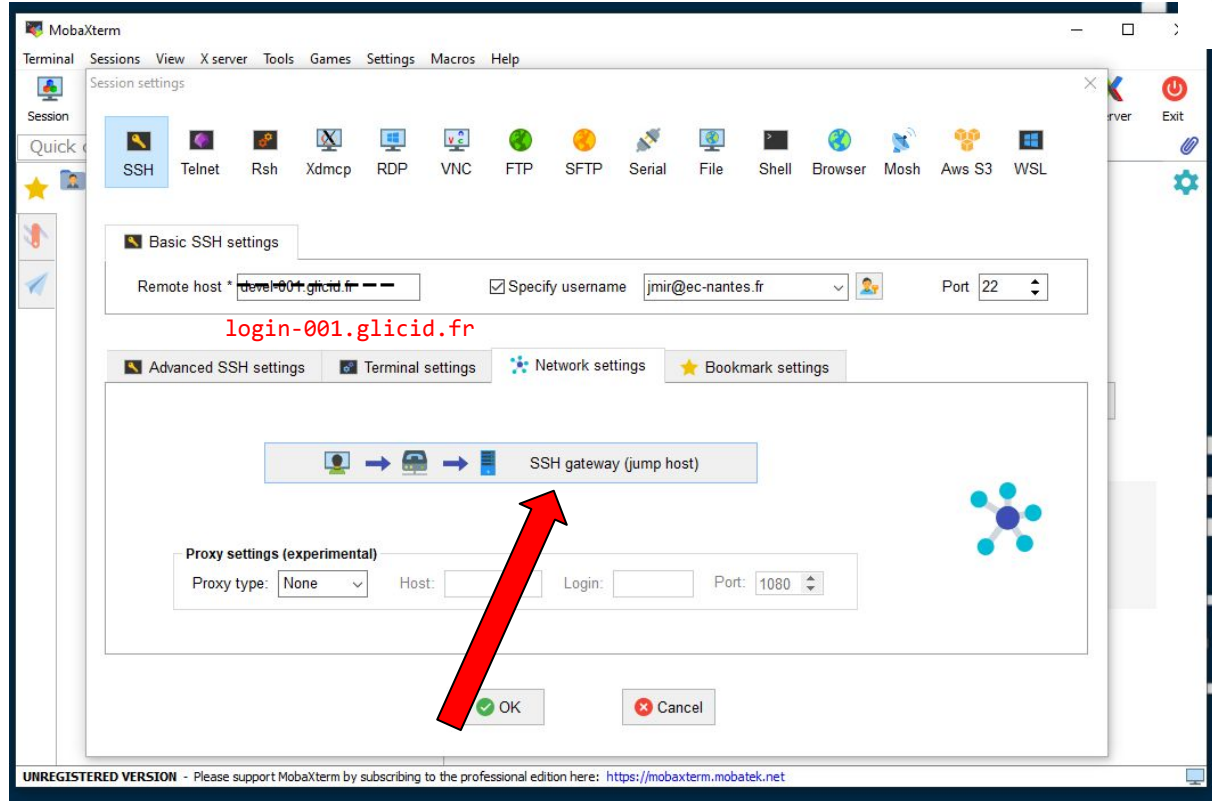
- Advanced SSH
  - Upload id\_rsa



# SSH configuration on Windows



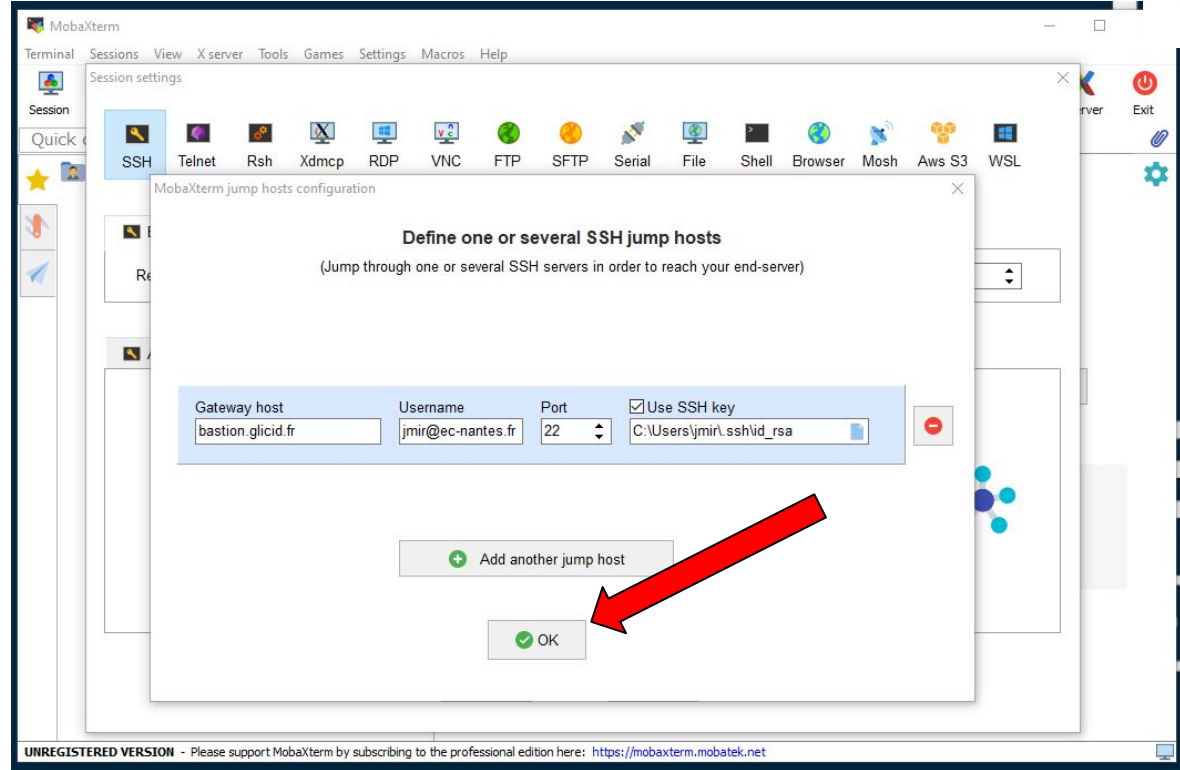
- SSH Gateway





# SSH configuration on Windows

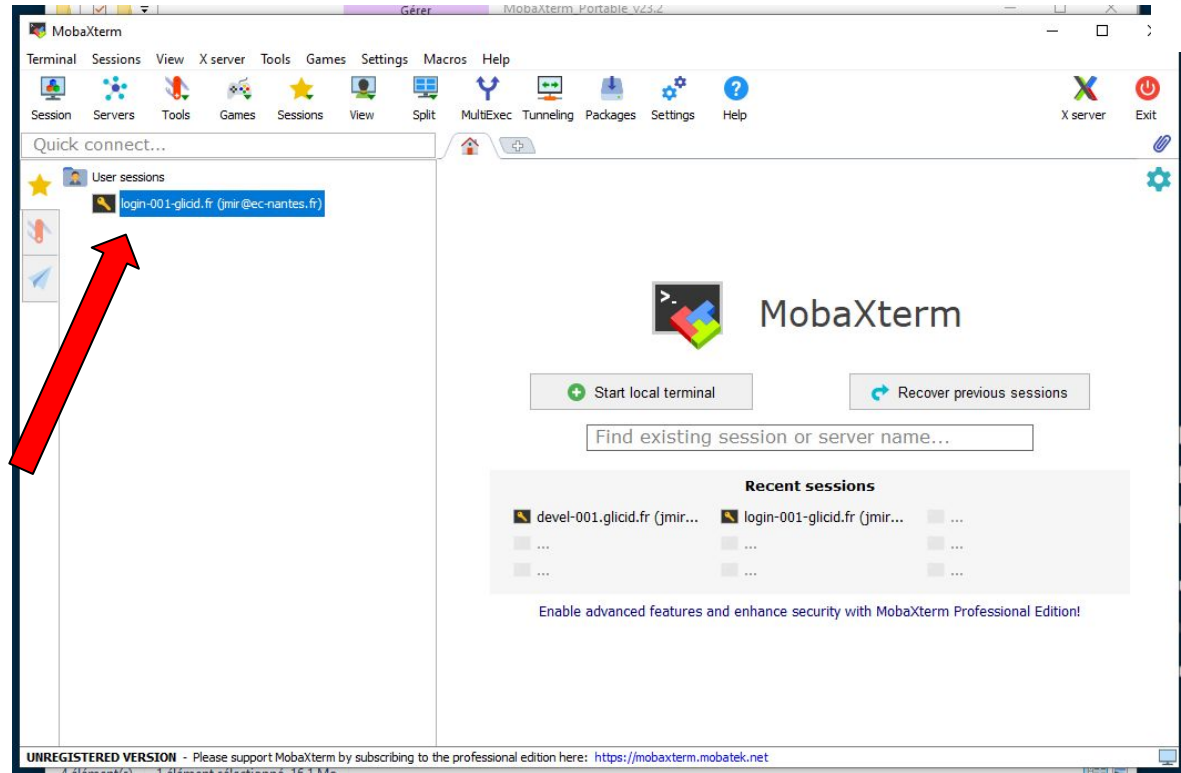
- SSH Gateway
  - Upload id\_rsa



# SSH configuration on Windows



- Execute Session





# SSH configuration on Windows

- Here you go...

The screenshot shows the MobaXterm interface. On the left, a file explorer displays the directory structure of the user's home folder. The main terminal window shows the output of an SSH session to a remote host.

```
login-001-glicid.fr (jmir@ec-nantes.fr)
Terminal Sessions View X server Tools Games Settings Macros Help
Session Servers Tools Games Sessions View Split MultiExec Tunneling Packages Settings Help
Quick connect...
/home/jmir@ec-nantes.fr/
Name Size (KB) Last modified
..
.cache 2023-09
.conda 2023-09
.config 2023-09
.local 2023-09
.mamba 2023-09
.slurm 2023-08
.ssh 2023-09
data 2023-09
test-containers 2023-07
test-containers-win 2023-09
testdir 2023-09
.bash_history 11 2023-09
.bashrc 1 2023-09
.condarc 1 2023-09
.guix-profile 1 2023-08
.jmir-gitlab 1 2023-09
.lessht 1 2023-09
.profile 1 2023-09
.python_history 1 2023-09
.viminfo 20 2023-09
.gitignore 1 2023-09
Remote monitoring
Follow terminal folder
UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: https://mobaxterm.mobatek.net
```

```
• MobaXterm Personal Edition v23.2 •
(SSH client, X server and network tools)
▶ SSH session to jmir@ec-nantes.fr@login-001.glicid.fr
• SSH gateway : ✓ (jmir@ec-nantes.fr@bastion.glicid.fr)
• SSH compression : ✓
• SSH-browser : ✓
• X11-forwarding : ✓ (remote display is forwarded through SSH)
▶ For more info, ctrl+click on help or visit our website.

Last login: Tue Sep 26 15:26:30 2023 from 10.50.255.250
jmir@ec-nantes.fr@guix-devel-001 ~$
```



# WINDOWS: File Management

- File Management
  - Just drag and drop
  - Or using upload option
  - Inside the terminal, it's Unix
  - So if you didn't pay interest in the command line session :(

# Get Your Hands Dirty

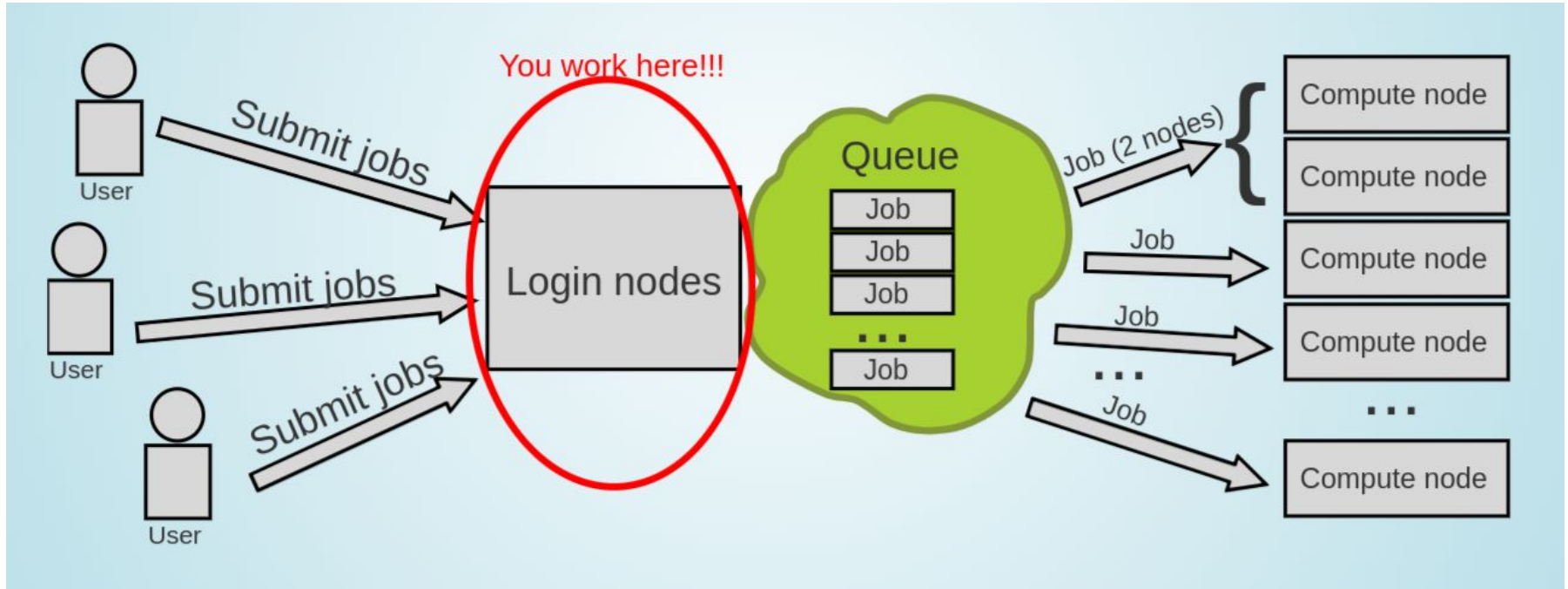




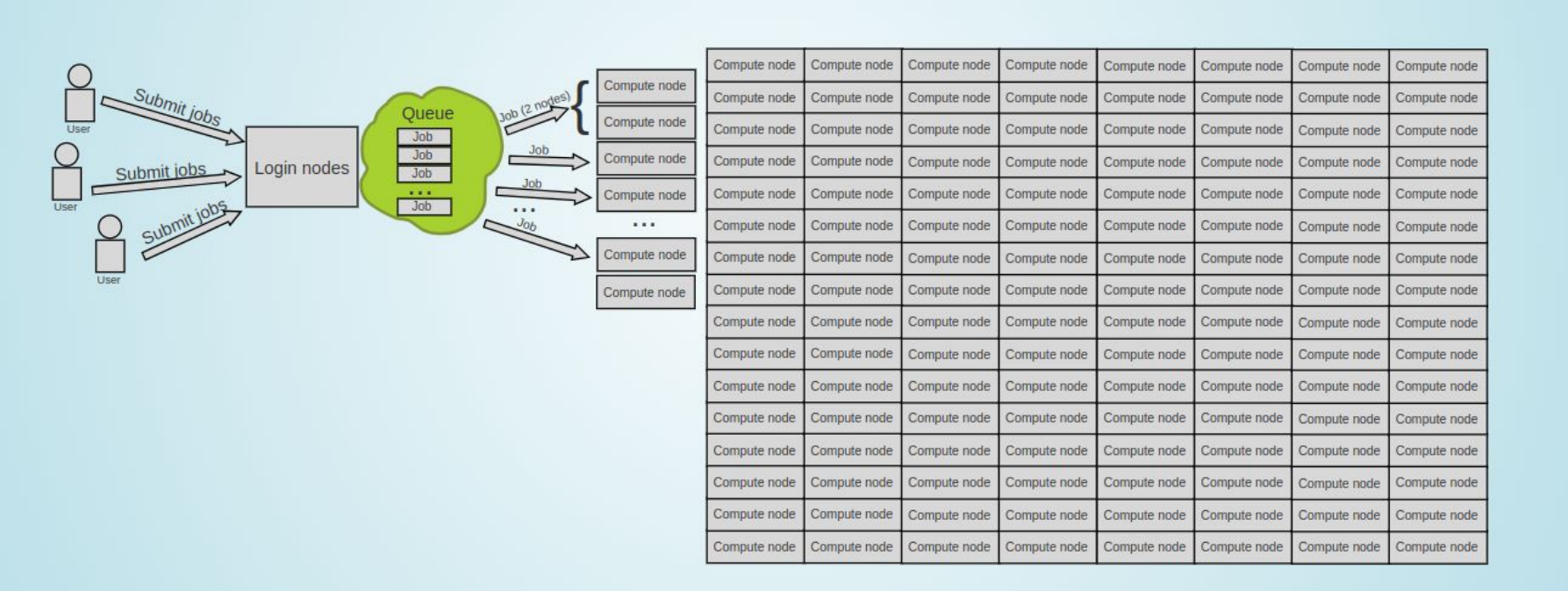
# Bringing Order To Chaos



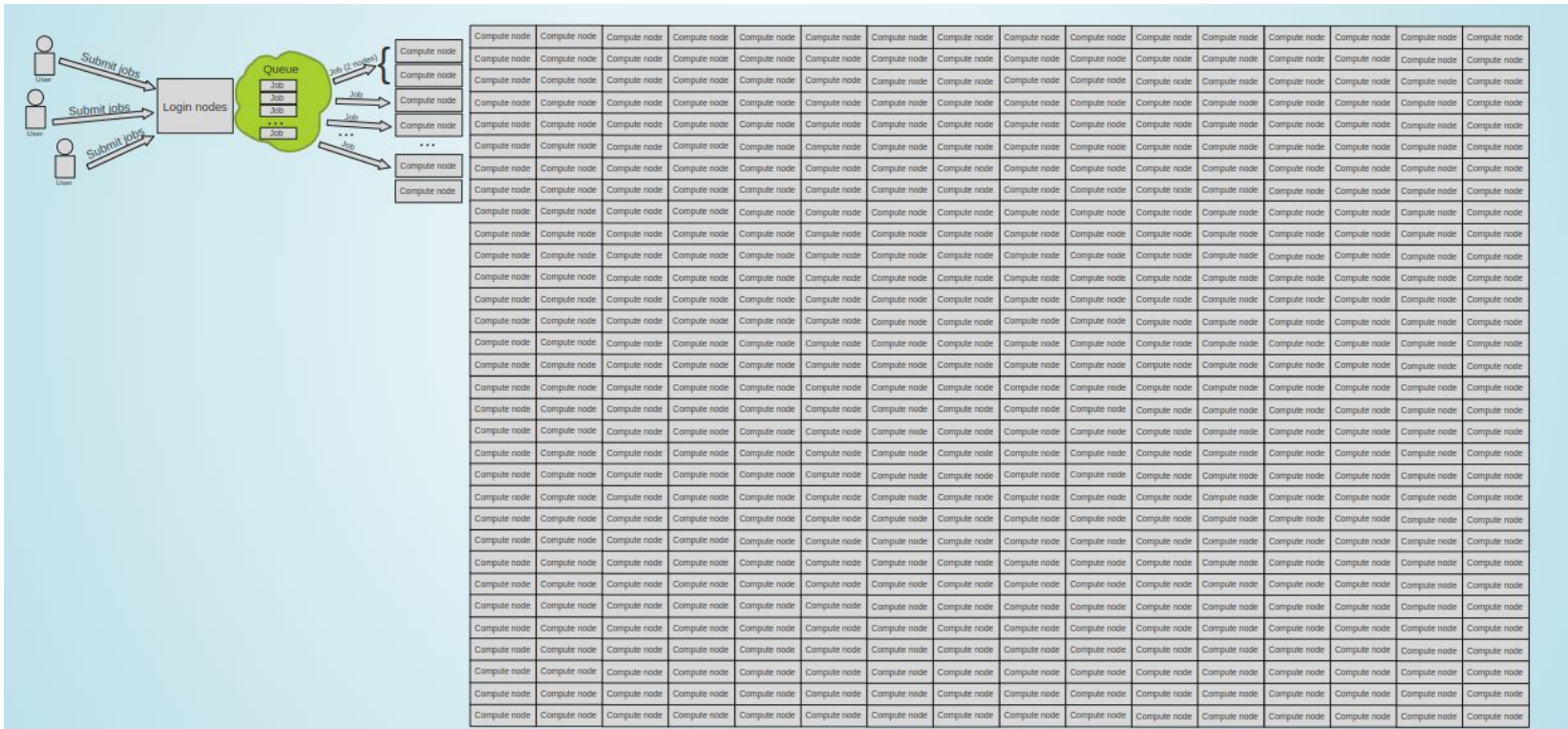
# You submit jobs



# But you don't use the whole Supercomputer



# There are many more users

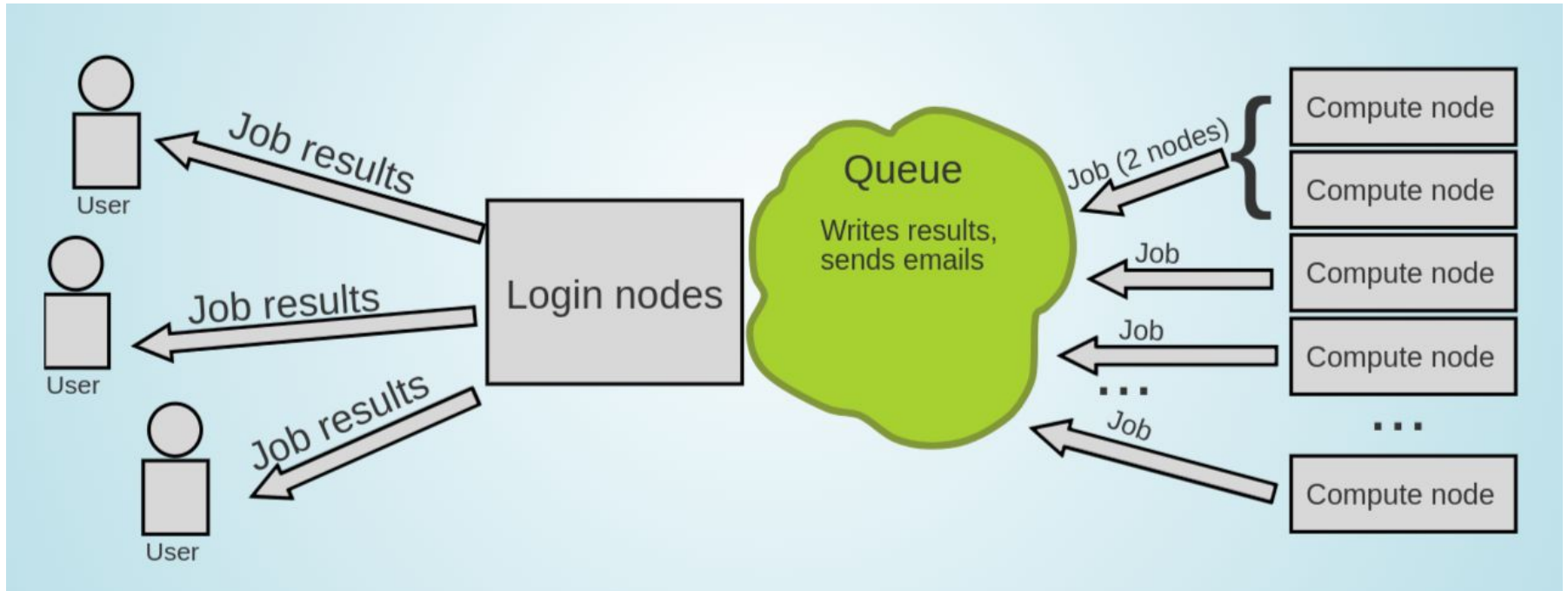


# Enter the queue, and wait

- Your job(s) enter the queue, and wait for its turn
- When there are enough resources for that job, it runs



# Results



## This it how it works

- User submits jobs
  - Job enters the queue
  - When it can, it runs
  - Sends results back to user
- 
- CAUTION
  - Login nodes are for submitting jobs, move files, compile, etc
  - NOT FOR TRAINING NEURAL NETS

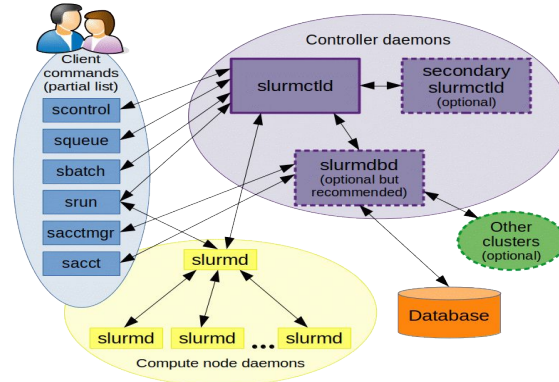
# Who will manage this workload?

- Need software that will distribute the jobs appropriately and manage the resources
- Keeps track of what nodes are busy/available, and what jobs are queued or running
- Tells the resource manager when to run which job on the available resources



# SLURM - Workload Manager

- **Simple Linux Utility for Resource Management (SLURM)**
- Open source, fault-tolerant, and highly scalable cluster management and job scheduling system for large and small Linux clusters
- It has centralized manager, **slurmctld**, to monitor resources and work
- Each compute node has a **slurmd daemon**, which can be compared to a remote shell: it waits for work, executes that work, returns status, and waits for more work.





# Basic Slurm Commands

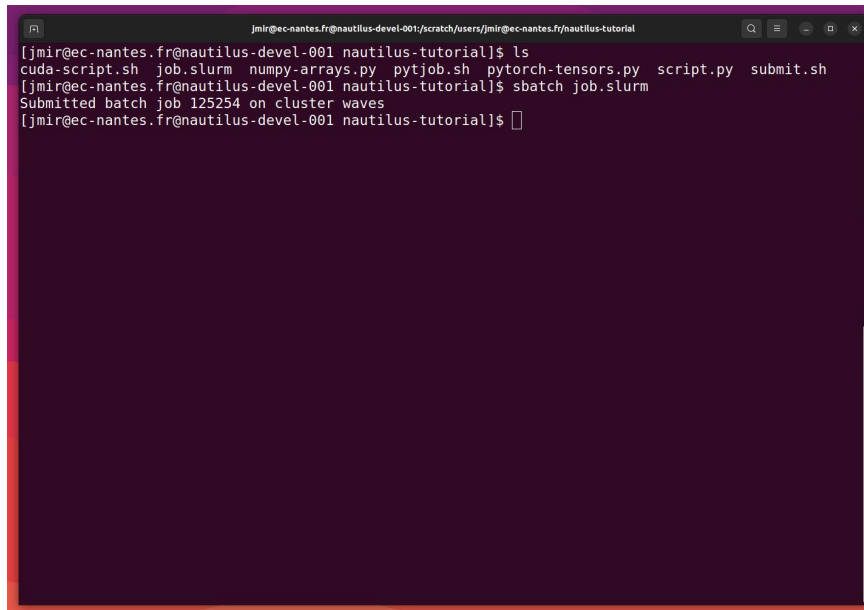


| Command              | Syntax  | Description   |
|----------------------|---|---|
| <code>sbatch</code>  | <code>sbatch &lt;job_id&gt;</code>                            | To submit job script for later execution (batch mode) |
| <code>sinfo</code>   | <code>sinfo</code>  | Get information about available nodes                 |
| <code>squeue</code>  | <code>squeue -u</code>  | Show information about jobs                           |
| <code>scancel</code> | <code>scancel &lt;job-id&gt;</code>                           | To terminate queued or running jobs                   |
| <code>srun</code>    | <code>srun</code><br><code>&lt;resource-parameters&gt;</code> | To run jobs interactively                             |
| <code>sacct</code>   | <code>sacct</code>  | Show information about current and previous jobs      |

# Basic Slurm Commands

- To submit a job

```
sbatch job.slurm
```



```
jmir@ec-nantes.fr@nautilus-devel-001/scratch/users/jmir@ec-nantes.fr/nautilus-tutorial
[jmir@ec-nantes.fr@nautilus-devel-001 nautilus-tutorial]$ ls
cuda-script.sh job.slurm numpy-arrays.py pytjob.sh pytorch-tensors.py script.py submit.sh
[jmir@ec-nantes.fr@nautilus-devel-001 nautilus-tutorial]$ sbatch job.slurm
Submitted batch job 125254 on cluster waves
[jmir@ec-nantes.fr@nautilus-devel-001 nautilus-tutorial]$
```

# Basic Slurm Commands

- Get information about available nodes

`sinfo`

```

[jmir@ec-nantes.fr@nautilus-devel-001: ~]$ sinfo
CLUSTER: nautilus
PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
standard up    infinite   1  inval cnode339
standard up    infinite   2  down* cnode[323,329]
standard up    infinite  10  drain cnode[302-308,337-338,340]
standard up    infinite   1  mix   cnode310
standard up    infinite   2  alloc cnode[301,309]
standard up    infinite  22  idle  cnode[312-322,325-328,330-336]
standard up    infinite   2  down  cnode[311,324]
bigmem  up    infinite   2  drain cnode[703-704]
bigmem  up    infinite   3  alloc cnode[701-702,706]
bigmem  up    infinite   3  idle  cnode[705,707-708]
gpu     up    infinite   4  mix   gnode[1-4]
visu    up    infinite   1  alloc visu1
visu    up    infinite   3  idle  visu[2-4]
all*    up    infinite   1  inval cnode339
all*    up    infinite   2  down* cnode[323,329]
all*    up    infinite  12  drain cnode[302-308,337-338,340,703-704]
all*    up    infinite   5  mix   cnode310,gnode[1-4]
all*    up    infinite   6  alloc cnode[301,309,701-702,706],visu1
all*    up    infinite  28  idle  cnode[312-322,325-328,330-336,705,707-708],visu[2-4]
all*    up    infinite   2  down  cnode[311,324]

CLUSTER: waves
PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
all*    up    9:00:00   1  unk*  budbud018
all*    up    9:00:00   1  mix   budbud020
all*    up    9:00:00   7  idle  budbud[014-017,019,021-022]
med     up    4-04:00:00  1  mix   budbud020
med     up    4-04:00:00  2  idle  budbud[021-022]
devel   up    20:00     1  unk*  vmworker-001
[jmir@ec-nantes.fr@nautilus-devel-001: ~]$

```

# Basic Slurm Commands

- To check Priority and MaxWall Time

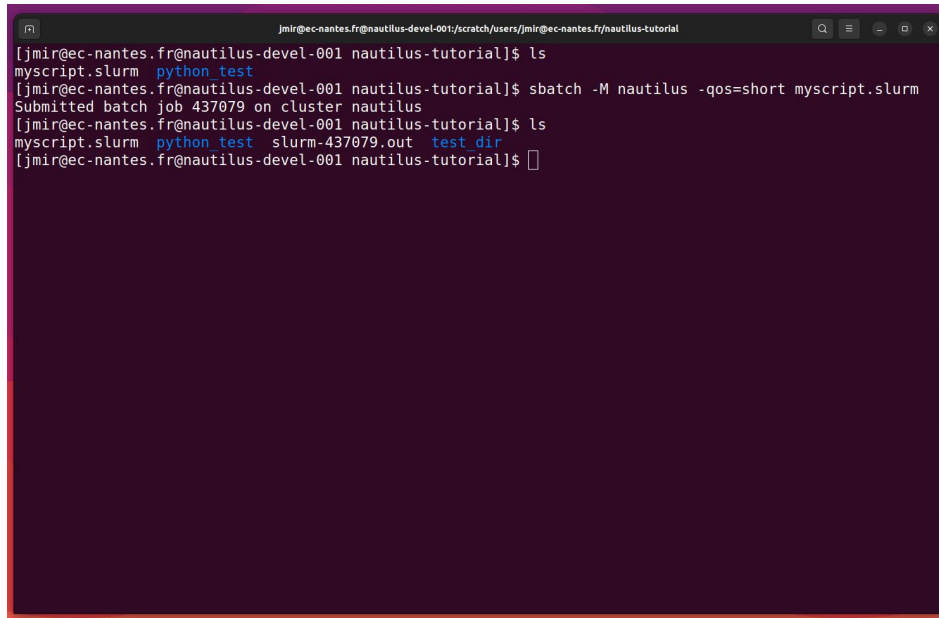
```
sacctmgr show qos format="name%20,priority,MaxJobsPerUser,MaxWall"
```

```
jmir@ec-nantes.fr@nautilus-devel-001:scratch/users/jmir@ec-nantes.fr/nautilus-tutorial
[jmir@ec-nantes.fr@nautilus-devel-001 nautilus-tutorial]$ sacctmgr show qos format="name%20,priority,MaxJobsPerUser,MaxWall"
-----
Name      Priority MaxJobsPU  MaxWall
-----
normal    1         1          00:05:00
short     50         1          1-00:00:00
medium    40         1          3-00:00:00
long      30         1          8-00:00:00
unlimited  10         1          00:00:00
debug     100        1          00:20:00
priority  200        1          8-00:00:00
[jmir@ec-nantes.fr@nautilus-devel-001 nautilus-tutorial]$
```

# Basic Slurm Commands

- Submit your slurm script

```
sbatch -M nautilus -p standard -q short <script-name>.slurm
```



```
jmir@ec-nantes.fr@nautilus-devel-001:~/scratch/users/jmir@ec-nantes.fr/nautilus-tutorial$ ls
myscript.slurm  python test
[jmir@ec-nantes.fr@nautilus-devel-001 nautilus-tutorial]$ sbatch -M nautilus -qos=short myscript.slurm
Submitted batch job 437079 on cluster nautilus
[jmir@ec-nantes.fr@nautilus-devel-001 nautilus-tutorial]$ ls
myscript.slurm  python test  slurm-437079.out  test_dir
[jmir@ec-nantes.fr@nautilus-devel-001 nautilus-tutorial]$
```

# Slurm - Batch Script



Sample script to run python code using conda environment

```
#!/bin/bash
#SBATCH --job-name=myjob          # create a short name for your job
#SBATCH --nodes=1                 # node count
#SBATCH --ntasks=1               # total number of tasks across all nodes
#SBATCH --cpus-per-task=1        # cpu-cores per task
#SBATCH --mem-per-cpu=2G         # memory per cpu-core
#SBATCH --gres=gpu:4             # number of gpus per node
#SBATCH --time=00:05:00         # total run time limit (HH:MM:SS)

cd /scratch/user/<username>      # go to your working directory / optional

hostname

python myscript.py
```

# Software Modules

# Software Modules

- Modules
  - Lot of useful software packages
  - Different versions
  - Maintained by experts
  - Optimized for the architecture
  - Users cannot install a module
  - Have to request the administrator



# How to use Modules?

- Useful commands

| Command  | Description                           |
|--|---------------------------------------|
| <code>module avail</code>                            | List modules                          |
| <code>module avail &lt;package_name&gt;</code>       | List all installed versions of python |
| <code>module load &lt;package_name&gt;</code>        | Load the default python version       |
| <code>module load &lt;package_name/3.11.5&gt;</code> | Load a specific version of python     |
| <code>module unload &lt;package_name&gt;</code>      | Unload python                         |
| <code>module list</code>                             | List currently loaded modules         |

# How to use Modules?

\$module avail

```
jmir@ec-nantes.fr@nautilus-devel-001:~$ module avail
----- /usr/share/Modules/modulefiles/applications -----
castem/2021 castem/2023 gaussian/g16-revA01 hyperworks/2022.2 lammps/15Jun2023 turbomole/7.41
----- /usr/share/Modules/modulefiles/libraries -----
aocl-blis/4.0 intel/ccl/2021.9.0 intel/dpl/latest intel/mkl32/2023.1.0
boost/1.82.0_gnu intel/ccl/latest intel/intel_ipp_ia32/2021.8.0 intel/mkl32/latest
cuda/12.2.0_535.54.03 intel/dnnl-cpu-gomp/2023.1.0 intel/intel_ipp_ia32/latest intel/tbb/2021.9.0
fftw/3.3.10_intel_serial intel/dnnl-cpu-gomp/latest intel/intel_ipp_intel64/2021.8.0 intel/tbb/latest
fftw/3.3.10_intel_serial_sp intel/dnnl-cpu-iomp/2023.1.0 intel/intel_ipp_intel64/latest intel/tbb32/2021.9.0
fftw/intelmpi/3.3.10_intel_intelmpi intel/dnnl-cpu-iomp/latest intel/intel_ippcp_ia32/2021.7.0 intel/tbb32/latest
fftw/omp/3.3.10_intel_omp intel/dnnl-cpu-tbb/2023.1.0 intel/intel_ippcp_ia32/latest libtool/2.4.6_gnu
gmsl/4.11.1_gnu intel/dnnl-cpu-tbb/latest intel/intel_ippcp_intel64/2021.7.0 netcdf/c-4.9.2_gnu
hdf5/1.14.1-2_gnu intel/dnnl/2023.1.0 intel/intel_ippcp_intel64/latest netcdf/f-4.6.1_gnu
hdf5/1.14.1-2_intel intel/dnnl/latest intel/mkl/2023.1.0 rdma/46.0_gnu
hdf5/intelmpi/1.14.1-2_intel_intelmpi intel/dpl/2022.1.0 intel/mkl/latest ucx/1.14.1_gnu
----- /usr/share/Modules/modulefiles/compilers -----
amd/4.0.0 intel/compiler-rt/2023.1.0 intel/compiler-rt32/latest intel/compiler32/2023.1.0 intel/icc/latest
cmake/3.26.4 intel/compiler-rt/latest intel/compiler/2023.1.0 intel/compiler32/latest intel/icc32/2023.1.0
gcc/13.1.0 intel/compiler-rt32/2023.1.0 intel/compiler/latest intel/icc/2023.1.0 intel/icc32/latest
----- /usr/share/Modules/modulefiles/tools -----
guix/latest intel/clck/latest intel/debugger/latest intel/dpct/latest intel/inspector/latest intel/oclfpga/latest
intel/advisor/2023.1.0 intel/dal/2023.1.0 intel/dev-utilities/2021.9.0 intel/init_opencl/2023.1.0 intel/itac/2021.9.0 intel/vtune/2023.1.0
intel/advisor/latest intel/dal/latest intel/dev-utilities/latest intel/init_opencl/latest intel/itac/latest intel/vtune/latest
intel/clck/2021.7.3 intel/debugger/2023.1.0 intel/dpct/2023.1.0 intel/inspector/2023.1.0 intel/oclfpga/2023.1.0
----- /usr/share/Modules/modulefiles/parallel -----
intel/mpi/2021.9.0 intel/mpi/latest openmpi/ucx/4.1.5_gcc_8.5.0_ucx_1.14.1_rdma_46.0
jmir@ec-nantes.fr@nautilus-devel-001:~$
```

# Guix Package Manager



# What is Guix?

- Package building system/Package manager
- Works on GNU/Linux
- Allows each user to manage his/her own packages
  - without root privilege
  - without interfering with other users
- Easy creation of isolated environments with designated packages
  - useful for per-project dependency management





# Guix Package Manager

- Useful commands

| Command  | Description   |
|--|---|
| <code>guix pull</code>                         | You need to run this at least once(maybe weekly :p) |
| <code>guix search &lt;package_name&gt;</code>  | Look for a package to install                       |
| <code>guix install &lt;package_name&gt;</code> | To install a package                                |
| <code>guix remove &lt;package_name&gt;</code>  | To remove a package                                 |
| <code>guix package -l</code>                   | List of installed packages                          |

# How to use Guix?

`$guix package -l`

```
[jmir@ec-nantes.fr@nautilus-devel-001 ~]$ guix package -l
Generation 1  août 01 2023 15:53:10
+ graphviz 7.0.1 out /gnu/store/8ljq5ipy0gs6w69rjigz11vf01zqzaxh-graphviz-7.0.1

Generation 2  sept. 07 2023 16:33:51
+ python 3.10.7 out /gnu/store/3hnmbi6yyn06w4xz7wpsw5nzb12l6xjv-python-3.10.7

Generation 3  sept. 20 2023 09:35:38
+ conda 22.9.0 out /gnu/store/gxrlqm3gggz3jwz79kw11125i3lhicb9-conda-22.9.0

Generation 4  sept. 20 2023 09:37:35
+ python-numpy 1.23.2 out /gnu/store/kz02cd8dcmryb2fk3ylniky2z333yi2-python-numpy-1.23.2

Generation 5  sept. 20 2023 10:10:55

Generation 6  sept. 20 2023 11:52:44
+ python-pandas 1.4.4 out /gnu/store/pg68s6204m0sf7g085qa937wvc6nf65m-python-pandas-1.4.4

Generation 7  sept. 20 2023 12:58:54
+ python-pytorch 1.13.1 out /gnu/store/icwxw62i5xpfdczv4sih0f58qw9ah89-python-pytorch-1.13.1

Generation 8  sept. 20 2023 15:25:47
+ python-numba 0.56.4 out /gnu/store/1lnsljkzdwsvpclgkg37kqr7qp0vg4i-python-numba-0.56.4

Generation 9  sept. 20 2023 16:20:15
+ python-torchvision 0.15.2 out /gnu/store/gjmf484aq6kw6gfd02474n1lfiw2wvix-python-torchvision-0.15.2

Generation 10  sept. 20 2023 16:32:05
+ python-pytorch-lightning 2.0.2 out /gnu/store/v2hvv552j2ywrhx6vs8cyj51ldhx2hd-python-pytorch-lightning-2.0.2

Generation 11  sept. 22 2023 09:48:01 (current)
+ conda 22.9.0 out /gnu/store/vf3j4n995jk9zh8yzbvm0mmwajdq71fx-conda-22.9.0
- conda 22.9.0 out /gnu/store/gxrlqm3gggz3jwz79kw11125i3lhicb9-conda-22.9.0

[jmir@ec-nantes.fr@nautilus-devel-001 ~]$
```

# Data Management

# Data management

- HOME (Personal Space/But don't train your neural network here)
- SCRATCH (Train it here)
  - HDD
  - SDD
- LAB-DATA
  - Users
  - Projects

```
jmir@ec-nantes.fr@nautilus-devel-001:LAB-DATA/GLiCID/projects
[jmir@ec-nantes.fr@nautilus-devel-001 ~]$ pwd
/home/jmir@ec-nantes.fr
[jmir@ec-nantes.fr@nautilus-devel-001 ~]$ cd /scratch/users/jmir@ec-nantes.fr/
[jmir@ec-nantes.fr@nautilus-devel-001 jmir@ec-nantes.fr]$ pwd
/scratch/users/jmir@ec-nantes.fr
[jmir@ec-nantes.fr@nautilus-devel-001 jmir@ec-nantes.fr]$
[jmir@ec-nantes.fr@nautilus-devel-001 jmir@ec-nantes.fr]$ cd
[jmir@ec-nantes.fr@nautilus-devel-001 ~]$ cd /LAB-DATA/
[jmir@ec-nantes.fr@nautilus-devel-001 LAB-DATA]$ ls
BiRD CEISAM GLiCID LS2N OSUNA
[jmir@ec-nantes.fr@nautilus-devel-001 LAB-DATA]$ cd GLiCID/
[jmir@ec-nantes.fr@nautilus-devel-001 GLiCID]$ ls
projects users
[jmir@ec-nantes.fr@nautilus-devel-001 GLiCID]$ cd projects/
[jmir@ec-nantes.fr@nautilus-devel-001 projects]$ ls
CLAM FI2309191 GenoBiRD Icitech MesoScaleABL
EnVision GCP GLiCID-admins ITX nuts-workshop
[jmir@ec-nantes.fr@nautilus-devel-001 projects]$
```



# Get Your Hands Dirty Again



# Thank you. Any questions?



Please answer the survey if you haven't yet  
<https://forms.gle/B4dto4axGm4EVPwaA>

## Useful links:

User Doc: <https://doc.glicid.fr>

Support: <https://help.glicid.fr> or [support@glicid.fr](mailto:support@glicid.fr)

Chat: On CLAM website

Admins: [tech@glicid.fr](mailto:tech@glicid.fr)

Forum: Coming soon

Status page: <https://ckc.glicid.fr>