



# Uni.lu HPC School 2020

## PS1: Preliminaries (SSH - OpenOnDemand)



Uni.lu High Performance Computing (HPC) Team

T. Valette and A. Olloh

University of Luxembourg (UL), Luxembourg

<http://hpc.uni.lu>



LUXEMBOURG  
LET'S MAKE IT HAPPEN



## Latest versions available on Github:



UL HPC tutorials:

<https://github.com/ULHPC/tutorials>

UL HPC School:

<http://hpc.uni.lu/hpc-school/>

PS1 tutorial sources:

<ulhpc-tutorials.rtfid.io/en/latest/preliminaries>



# Summary

## 1 Introduction

## 2 Preliminaries Setup on your Laptop/Workstation

System Environment Adaptation

Distributed Version Control with Git

## 3 SSH Secure Shell

Installation

Usage

Data transfer using SSH

Prepare your laptop for next Practical Sessions

## 4 Discover Open OnDemand

## Main Objectives of this Session



- Review **recommended setup for your personnal Laptop**
  - ↪ Windows, Mac OS or Linux
- Understand **SSH** (Secure SHell)
- **Review SSH configuration** allowing to connect to the **UL HPC Platform**
  - ↪ SSH configuration
  - ↪ Generate your SSH key pair
  - ↪ Upload your SSH key on ULHPC Identity Management Portal
- **Discover ULHPC Open OnDemand**
  - ↪ HPC Web portal
  - ↪ Job composer

**OPEN**  **OnDemand**

## Vocabulary related to HPC (1/2)

**Compute node** physical server on which we run the computation (your code)

**Cluster** group of compute nodes interconnected to each others

**Processor/CPU** Central Processing Unit usually refers to a processor, chip of the server that process the instructions of the program

**Core** 1 processor chip usually contains several CPUs named cores

**GPU** Graphics Processing Unit, chip designed for image processing and computer graphics

## Vocabulary related to HPC (2/2)

**Resources** Every component of the cluster that you have access. Can refer to CPU, core, memory, network switch...

**Job** Allocation resources for a specific user and a specific amount of time

**Reservation** Allocate a job in the future, in advance in respect with rules (priority, job type...)

**Walltime** Maximum time allocated for a specific job

**Job Scheduler** Software that schedule all the jobs according to their priority.

**Job queue** Before being scheduled, jobs are waiting in a queue for being processed by the scheduler

**Partition** Set of resources (nodes) with the same policies applied to it

# Summary

## 1 Introduction

## 2 Preliminaries Setup on your Laptop/Workstation

System Environment Adaptation

Distributed Version Control with Git

## 3 SSH Secure Shell

Installation

Usage

Data transfer using SSH

Prepare your laptop for next Practical Sessions

## 4 Discover Open OnDemand

## Preliminaries

ULHPC Tutorials: Setup Preliminaries

[ulhpc-tutorials.rtfd.io/en/latest/setup/preliminaries](http://ulhpc-tutorials.rtfd.io/en/latest/setup/preliminaries)

- Pre-configuration expected on your laptop
- Online Accounts, at least:
  - Uni.lu HPC - read and accept the Acceptable Use Policy (AUP) 2.0
  - Github - recommended professional login: <letter><name> (Ex: svarrette)
  - Vagrant Cloud
  - Docker Hub
- All instructions/practical sessions guided against a Unix/Linux system
  - Native OS within **all** HPC facilities
  - interaction from any OS (Windows, Mac OS or Linux) is possible for **all** operations
    - ✓ ... unlike WebEx Training/Event

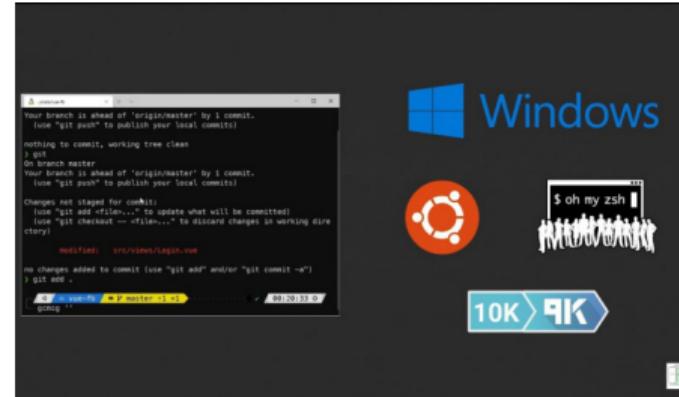
## Recommended Software List

Platform	Software	Description	Usage
Mac OS	Homebrew	The missing package manager for macOS	<code>brew install ...</code>
Mac OS	iTerm2	enhanced Terminal	
Windows	Chocolatey	Package Manager for Windows	<code>choco install ...</code>
Windows	Windows Subsystem for Linux (WSL)	Emulation-like translation of Linux kernel system calls	
Windows	Ubuntu over WSL	Linux Ubuntu on Windows	
Windows	Windows Terminal		
Windows	MobaXTERM	Terminal with tabbed SSH client	
Windows	SourceTree	enhanced git GUI	
Windows/Linux	Virtual Box	Free hypervisor provider for Vagrant	
Windows/Linux	Vagrant	Reproducible environments made easy.	
Windows	Docker for Windows	Lightweight Reproducible Containers	
Linux	Docker for Ubuntu	Lightweight Reproducible Containers	

# Preliminaries Setup for Windows

≥ Windows 10 release 1607

- New “**Windows Subsystem for Linux**” Ubuntu Linux binaries
  - can run directly on Windows without the need for the Linux Kernel running in a VM.
    - ✓ provides emulation-like translation of Linux kernel system calls to the Windows kernel
  - Windows **feature, requires to turn on Developer mode**
- Install Ubuntu within the **Microsoft Store**
  - You now have an Ubuntu terminal by running bash command
    - ✓ (**optional**) consider installing oh-my-zsh and the Powerlevel10k prompt - guide
- Install **Windows Terminal**
  - Setup prompt in Windows Terminal & PowerShell
  - **Now open Ubuntu profile**
    - ✓ enjoy an *fully-featured Linux terminal*
      - ... incl. native SSH
- (*also for OLD systems*) install **MobaXterm**



## Preliminaries Setup for Linux (incl. WSL Ubuntu) & Mac OS

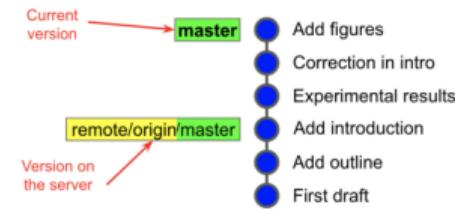
- Native SSH client available, accessible with your terminal
  - Make sure you can open and interact with a terminal on your computer
  - There exists **better** terminal applications than the default ones proposed on your system
    - ✓ Mac OS: iTerm2
    - ✓ Windows Terminal
    - ✓ Linux: Guake Terminal, Terminator
- If you already have an SSH Client, you should have this following output:

```
$(laptop)> ssh -V
OpenSSH_8.4p1, OpenSSL 1.1.1h 22 Sep 2020
```

- **(optional)** If you want a fancy et nice terminal layout, consider using zsh and oh-my-zsh
  - [guide](#) for Linux
  - [guide](#) for Mac OS

## Git in a Nutshell

- **Distributed Version Control System**
  - ↪ Manage changes in documents and Keep the history
- **Synchronization with a server (optional)**
  - ↪ Collaborative work
  - ↪ Backup of your documents/code
- Well suited for **text** files (source code, LaTeX article etc.)
  - ↪ can also be used for binary files (images, HDF5, Office docs...)
    - ✓ if the size is reasonable
  - ↪ typically avoid to version temporary (scratch) files
  - ↪ extension exists to support large files (Git-LFS)



## Installation Notes

```
### Mac OS X, using Homebrew - https://brew.sh
brew install git git-gui git-flow gitk tig kdiff3
### Ubuntu / WSL / RedHat
{ sudo apt-get | yum } install git-core git-flow tig gitk kdiff3
### Windows, using Chocolatey - https://chocolatey.org/
$> choco.exe install git gitflow-avh
```

- **Windows:** Git for Windows: incl. Git Bash/GUI & Shell Integration
  - (**better**) Windows Subsystem for Linux (WSL)
    - ✓ Ubuntu for Windows within the Microsoft Store



## Installation Notes

```
### Mac OS X, using Homebrew - https://brew.sh
brew install git git-gui git-flow gitk tig kdiff3
### Ubuntu / WSL / RedHat
{ sudo apt-get | yum } install git-core git-flow tig gitk kdiff3
### Windows, using Chocolatey - https://chocolatey.org/
$> choco.exe install git gitflow-avh
```

- **Windows:** Git for Windows: incl. Git Bash/GUI & Shell Integration
  - (**better**) Windows Subsystem for Linux (WSL)
    - ✓ Ubuntu for Windows within the Microsoft Store

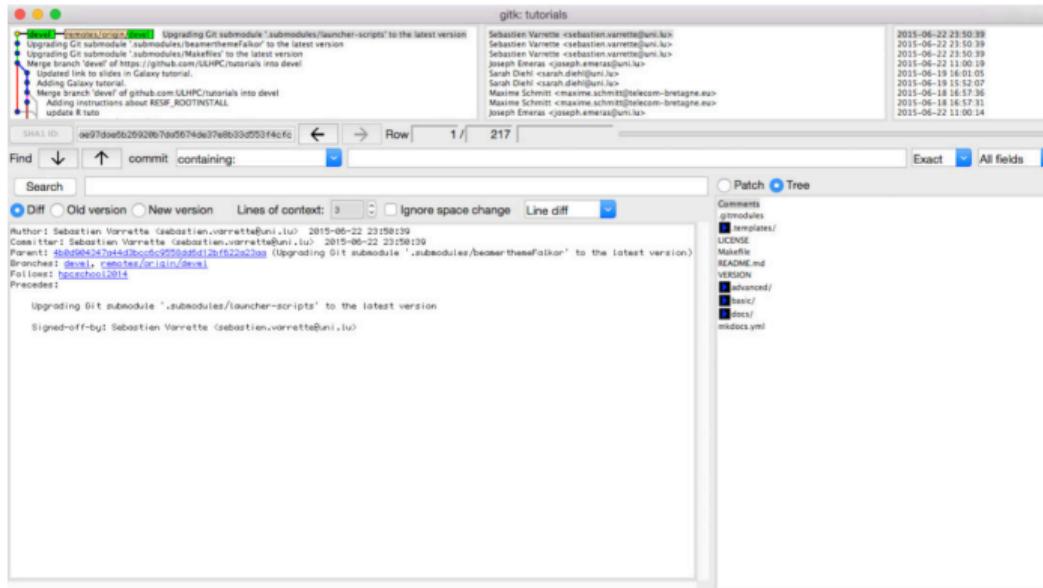
Your Turn!

- Ensure you have git installed on your system

git --version

# Other related tools

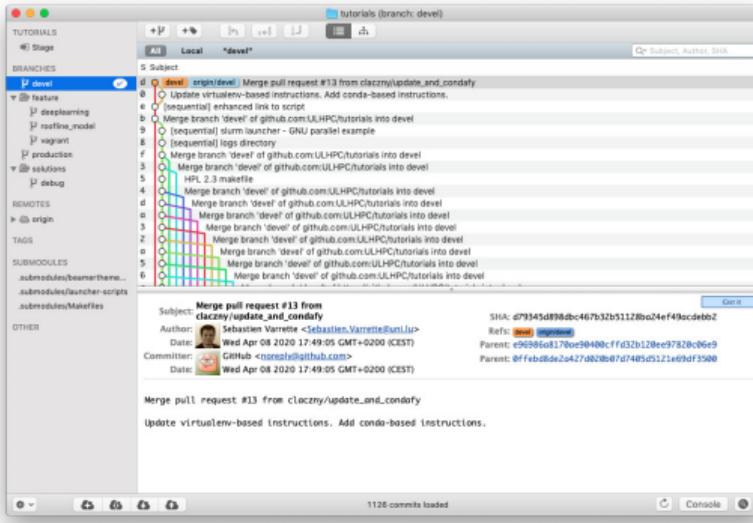
- Git GUI (default): **GitK**



# Other related tools

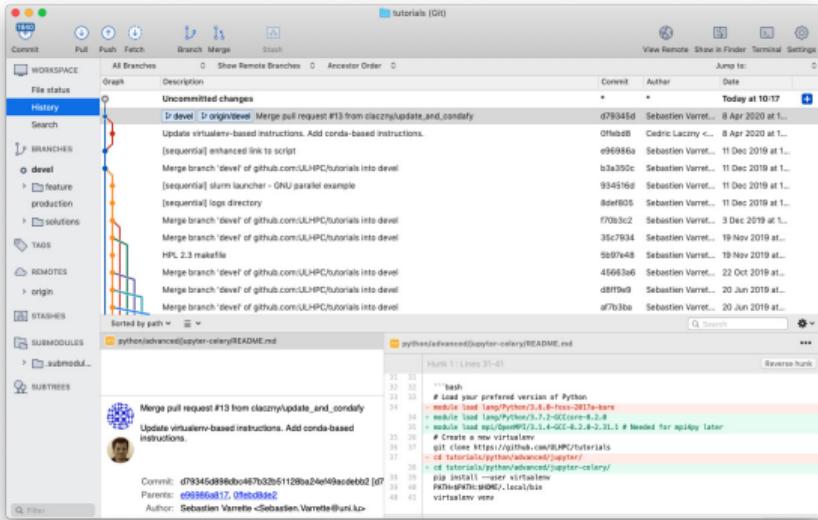
- Git GUI (Mac OS): [GitX-dev](#)

<http://rowanj.github.io/gitx/>



# Other related tools

- Git GUI (Windows/Mac OS): **SourceTree**  
→ Alternatives: <https://git-scm.com/downloads/guis/>



## Preliminary Configurations

- You **SHOULD at least** configure your name and email to commit
  - ↪ **(optional but useful)** inform Git of your GPG signing key

```
### Basic Git defaults - open a Terminal
git config --global user.name "Firstname LastName"
git config --global user.email "<email>@<domain>"
git config --global user.signingkey <gpg-keyID>
git config --global color.ui true
# Check that the changes are effective (see also ~/.gitconfig or ~/.config/git/config)
git config -l | grep user
```

## Preliminary Configurations

- You **SHOULD at least** configure your name and email to commit
  - ↪ **(optional but useful)** inform Git of your GPG signing key

```
### Basic Git defaults - open a Terminal
git config --global user.name "Firstname LastName"
git config --global user.email "<email>@<domain>"
git config --global user.signingkey <gpg-keyID>
git config --global color.ui true
# Check that the changes are effective (see also ~/.gitconfig or ~/.config/git/config)
git config -l | grep user
```

### Hands-on Git Config

▶ url ◀ | [github](#) | [src](#)

- **Git** Installation and useful derived tools
- **Git Initial Setup**

git-flow, tig, kdiff3, meld...  
git config --global [...]

## Shell Integration: Bash prompt

For **experienced users** only!

- Bash Prompt integration: `_git_ps1()` in your PS1 variable  
↪ See [contrib/completion/git-prompt.sh](#) (Git Sources)

```
~/src/libgit2 (development *)$ █
```

## Shell Integration: Bash prompt

For experienced users only!

- Bash Prompt integration:
  - ↪ See [contrib/completion/git-prompt.sh](#) (Git Sources)
- OR **Agnoster Theme for Bash**
  - ↪ requires a Powerline-patched font

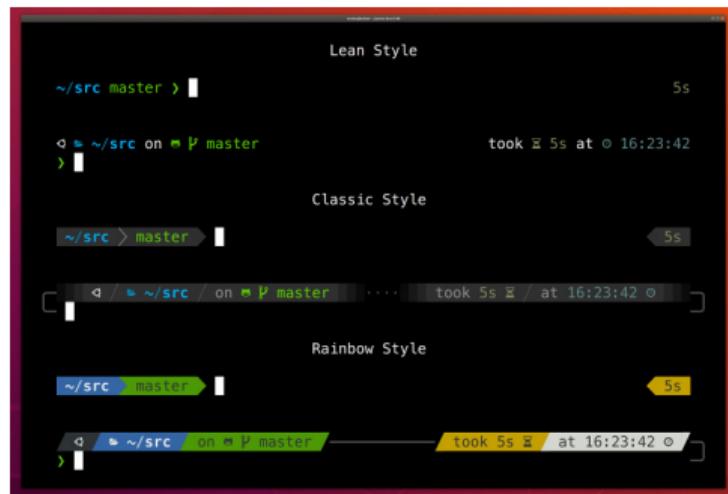
```
~ cd testproject
~/testproject p master gco detached-head-state -q
~/testproject - fdffaf6 touch dirty-working-directory
~/testproject - fdffaf6± cd
~ ssh milly
Welcome to Ubuntu 11.04 (GNU/Linux 2.6.18-308.8.2.el5.028stab101.1 x86_64)
Last login: Wed Sep 26 03:42:49 2012 from 71-215-222-90.mpls.qwest.net
agnoster@milly ~
Connection to milly.agnoster.net closed.
~ sudo -s
Password:
$ root@Arya ~ top &
[1] 34523
[1] + 34523 suspended (tty output) top
$ root@Arya ~ rm no-such-file
rm: no-such-file: No such file or directory
x $ root@Arya ~ kill %
[1] + 34523 terminated top
$ root@Arya ~
~
```

## Shell Integration: ZSH

For **experienced users** only!



- Oh my ZSH!:
  - community-driven framework
  - manages your Zsh configuration
  - multiple **plugins**, (super nice) **themes**
    - ✓ Ex: agnoster theme / powerline
  - Install: <https://ohmyz.sh/#install>
- Consider Powerlevel10k theme for ZSH



The terminal window displays three different ZSH themes:

- Lean Style**: Shows a simple black background with white text. The command run is `q ~ ~/src on P master` and it took 5s at 16:23:42.
- Classic Style**: Shows a black background with white text. The command run is `q ~ ~/src > master` and it took 5s at 16:23:42.
- Rainbow Style**: Shows a black background with colored text (green, blue, yellow). The command run is `q ~ ~/src > master` and it took 5s at 16:23:42.

## Shell Integration: Powershell

- You probably want to install Windows Terminal
  - ↪ Install the Windows Terminal from the Microsoft Store.
  - ↪ Install the Cascadia Code PL font



## Shell Integration: Powershell

- You probably want to install Windows Terminal
  - ↪ Install the Windows Terminal from the Microsoft Store.
  - ↪ Install the Cascadia Code PL font

For experienced users only!

- Install posh-git and oh-my-posh:

```
# Install posh-git and oh-my-posh in (administrator) powershell
# If Error: "Running scripts is disabled on this system"
Get-ExecutionPolicy -List
Set-ExecutionPolicy RemoteSigned -Scope CurrentUser
# Install
Install-Module posh-git -Scope CurrentUser
Install-Module oh-my-posh -Scope CurrentUser
Set-Theme Paradox
```



# Summary

## 1 Introduction

## 2 Preliminaries Setup on your Laptop/Workstation

System Environment Adaptation

Distributed Version Control with Git

## 3 SSH Secure Shell

Installation

Usage

Data transfer using SSH

Prepare your laptop for next Practical Sessions

## 4 Discover Open OnDemand

## SSH: Secure Shell

- Ensure **secure** connection to remote (UL) server
  - ↪ establish **encrypted** tunnel using **asymmetric keys**
    - ✓ **Public** id\_rsa.pub vs. **Private** id\_rsa (**without** .pub)
    - ✓ typically on a non-standard port (**Ex:** 8022)
    - ✓ Basic rule: 1 machine = 1 key pair
  - ↪ the private key is **SECRET**: **never** send it to anybody
    - ✓ Can be protected with a passphrase

*limits kiddie script*

## SSH: Secure Shell

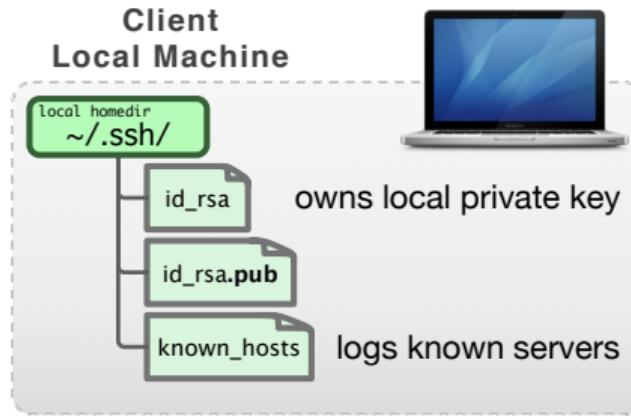
- Ensure **secure** connection to remote (UL) server
  - ↪ establish **encrypted** tunnel using **asymmetric keys**
    - ✓ **Public** id\_rsa.pub vs. **Private** id\_rsa (**without** .pub)
    - ✓ typically on a non-standard port (**Ex:** 8022)
    - ✓ Basic rule: 1 machine = 1 key pair
  - ↪ the private key is **SECRET**: **never** send it to anybody
    - ✓ Can be protected with a passphrase
- SSH is used as a secure backbone channel for **many** tools
  - ↪ Remote shell **i.e** remote command line
  - ↪ File transfer: **rsync**, **scp**, **sftp**
  - ↪ versionning synchronization (**svn**, **git**), **github**, **gitlab** etc.

*limits kiddie script*

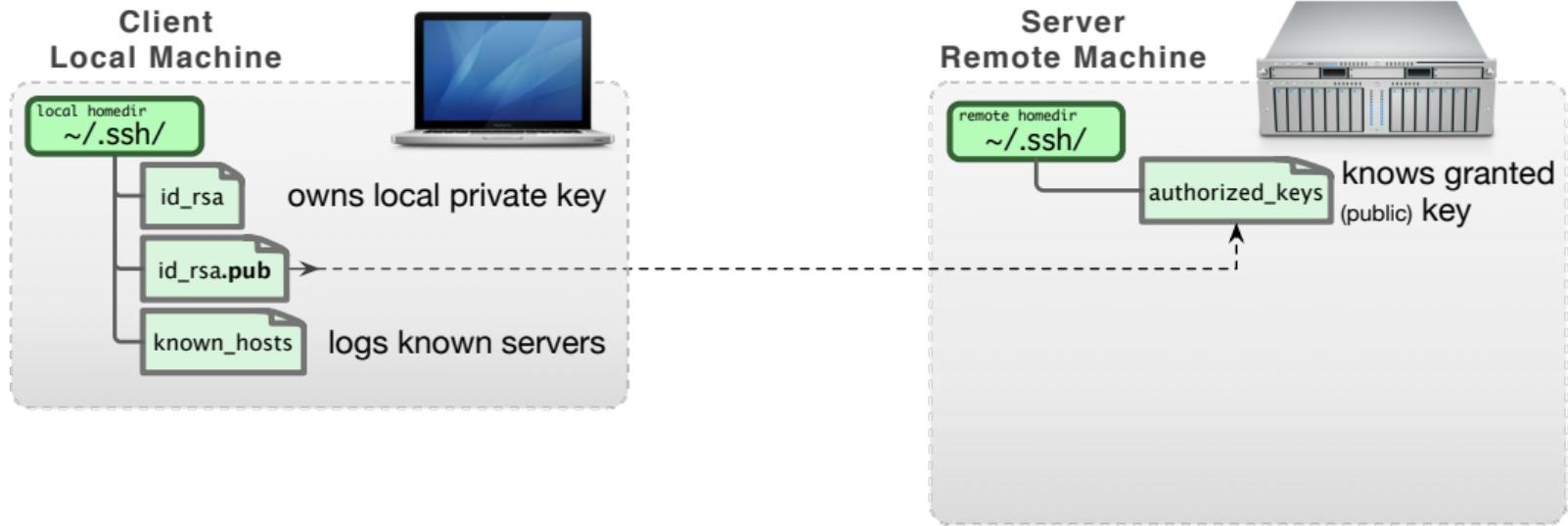
# SSH: Secure Shell

- Ensure **secure** connection to remote (UL) server
  - ↪ establish **encrypted** tunnel using **asymmetric keys**
    - ✓ **Public** id\_rsa.pub vs. **Private** id\_rsa (**without** .pub)
    - ✓ typically on a non-standard port (**Ex:** 8022)
    - ✓ Basic rule: 1 machine = 1 key pair
  - ↪ the private key is **SECRET**: **never** send it to anybody
    - ✓ Can be protected with a passphrase
- SSH is used as a secure backbone channel for **many** tools
  - ↪ Remote shell **i.e** remote command line
  - ↪ File transfer: **rsync**, **scp**, **sftp**
  - ↪ versionning synchronization (**svn**, **git**), **github**, **gitlab** etc.
- Authentication:
  - ↪ **password** (disable if possible)
  - ↪ **(better) public key authentication**

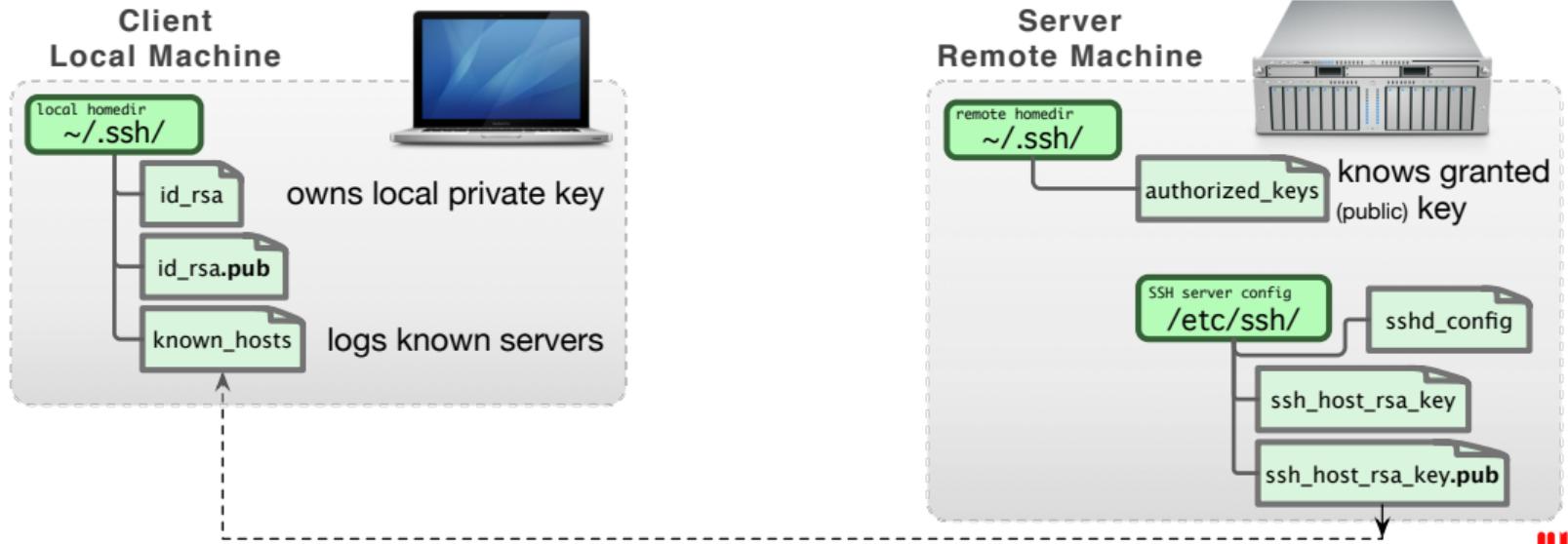
## SSH: Public Key Authentication



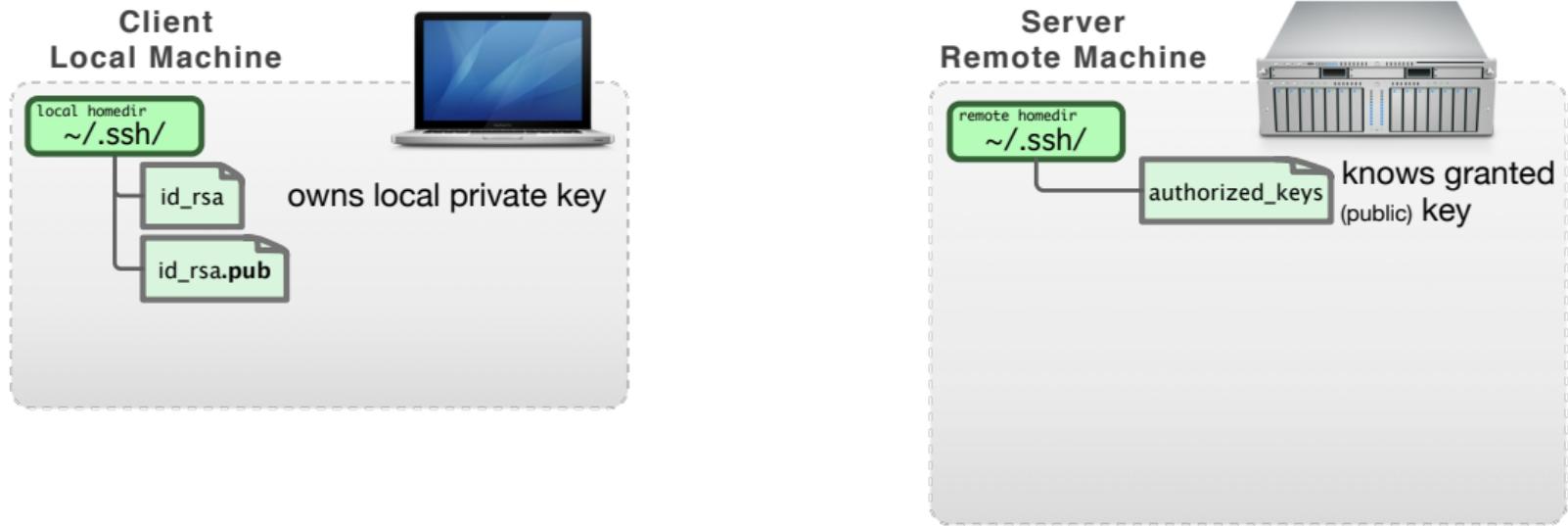
## SSH: Public Key Authentication



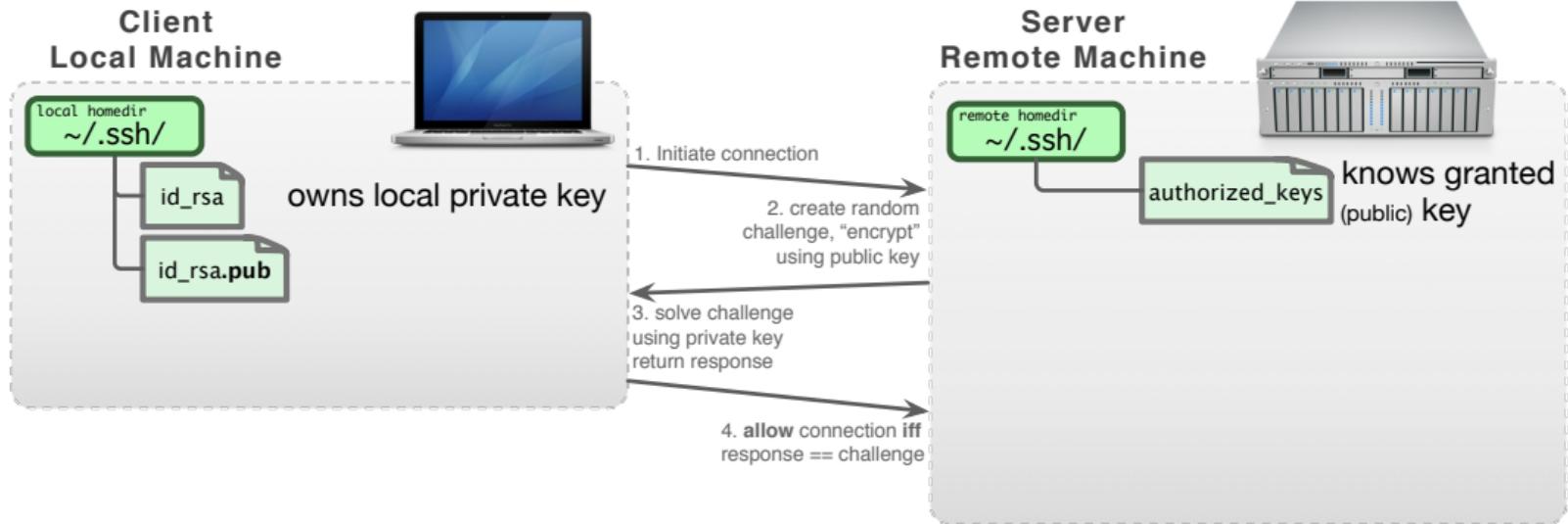
# SSH: Public Key Authentication



## SSH: Public Key Authentication



## SSH: Public Key Authentication



## SSH: Public Key Authentication

- **Restrict to public key authentication:** /etc/ssh/sshd\_config:

```
PermitRootLogin no
# Disable Passwords
PasswordAuthentication no
ChallengeResponseAuthentication no
```

```
# Enable Public key auth.
RSAAuthentication yes
PubkeyAuthentication yes
```

## SSH Setup on Linux / Mac OS

- OpenSSH natively supported; configuration directory : `~/.ssh/`
  - ↪ package `openssh-client` (Debian-like) or `ssh` (Redhat-like)
- SSH Key Pairs (public vs private) generation:
  - ↪ specify a **strong passphrase**
    - ✓ protect your **private** key from being stolen **i.e.** impersonation
    - ✓ **drawback:** passphrase must be typed to use your key

**ssh-keygen**

## SSH Setup on Linux / Mac OS

- OpenSSH natively supported; configuration directory : `~/.ssh/`
    - ↪ package `openssh-client` (Debian-like) or `ssh` (Redhat-like)
  - SSH Key Pairs (public vs private) generation:
    - ↪ specify a **strong passphrase**
      - ✓ protect your **private** key from being stolen **i.e.** impersonation
      - ✓ ~~drawback: passphrase must be typed to use your key~~
- ssh-keygen**
- ssh-agent**

## SSH Setup on Linux / Mac OS

- OpenSSH natively supported; configuration directory : `~/.ssh/`
  - ↪ package `openssh-client` (Debian-like) or `ssh` (Redhat-like)
- SSH Key Pairs (public vs private) generation:
  - ↪ specify a **strong passphrase**
    - ✓ protect your **private** key from being stolen **i.e.** impersonation
    - ✓ ~~drawback:~~ ~~passphrase must be typed to use your key~~

`ssh-keygen`

`ssh-agent`

DSA Keys + RSA 1024/2048 bits Dare deprecated now!

## SSH Setup on Linux / Mac OS

- OpenSSH natively supported; configuration directory : `~/.ssh/`
  - ↪ package `openssh-client` (Debian-like) or `ssh` (Redhat-like)
- SSH Key Pairs (public vs private) generation:
  - ↪ specify a **strong passphrase**
    - ✓ protect your **private** key from being stolen i.e. impersonation
    - ✓ ~~drawback: passphrase must be typed to use your key~~

`ssh-keygen`

`ssh-agent`

DSA Keys + RSA 1024/2048 bits Dare deprecated now!

```
$> ssh-keygen -t rsa -b 4096 -o -a 100 # 4096 bits RSA
```

```
(better) $> ssh-keygen -t ed25519 -o -a 100 # new sexy Ed25519
```

### Private (identity) key

`~/.ssh/id_{rsa,ed25519}`

### Public Key

`~/.ssh/id_{rsa,ed25519}.pub`

## SSH Setup on Windows: the OLD way

- Putty Suite, includes:
  - PuTTY, the free SSH client
  - Pageant, an SSH authentication agent for PuTTY tools
  - PLink, th PuTTy CLI and PuTTYgen, an RSA and DSA key generation utility

## SSH Setup on Windows: the OLD way

- Putty Suite, includes:
  - PuTTY, the free SSH client
  - Pageant, an SSH authentication agent for PuTTY tools
  - PLink, the PuTTy CLI and PuTTYgen, an RSA and DSA key generation utility

PuTTY ≠ OpenSSH

## SSH Setup on Windows: the OLD way

- Putty Suite, includes:
  - PuTTY, the free SSH client
  - Pageant, an SSH authentication agent for PuTTY tools
  - PLink, th PuTTy CLI and PuTTYgen, an RSA and DSA key generation utility

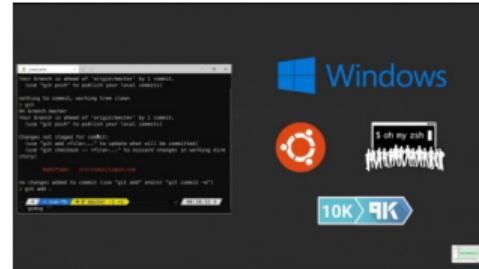
### PuTTY ≠ OpenSSH

- Putty keys are **NOT** supported by OpenSSH (yet can be exported)
- Binding Pageant with OpenSSH agent is **NOT** natively supported
  - Third-party tools like `ssh-pageant` are made for that
  - Combine nicely with `Git bash`
- with PLink, hostnames eventually refer to **PuTTY Sessions**
  - **NEVER** to SSH entries in `~/.ssh/config`
  - This usage might be hidden... Ex: `$GIT_SSH` etc.

<https://git-for-windows.github.io/>

# SSH Setup on Windows

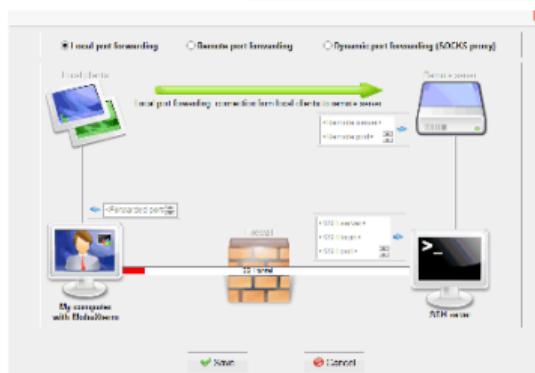
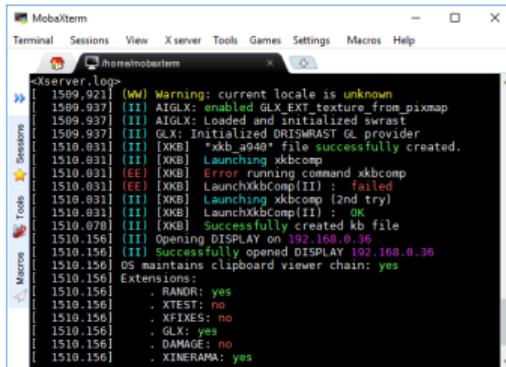
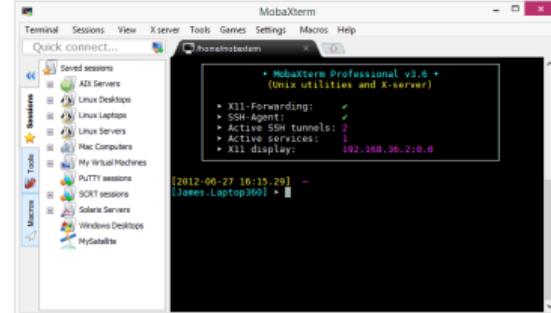
- New “**Windows Subsystem for Linux**” Ubuntu Linux binaries
  - ↪ Windows feature, requires to turn on Developer mode
  - ↪ Install Ubuntu within the [Microsoft Store](#)
  - ↪ You now have an Ubuntu terminal by running bash command
    - ✓ ([optional](#)) consider installing `oh-my-zsh` and the [Powerlevel10k](#) prompt - [guide](#)
- Install [Windows Terminal](#)
  - ↪ Setup prompt in Windows Terminal and PowerShell - [guide](#)
  - ↪ Now open Ubuntu profile to enjoy an fully-featured Linux terminal, incl. native SSH



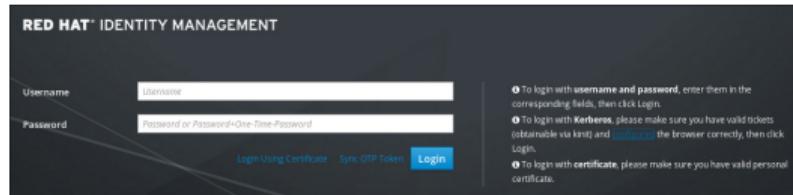
# SSH Setup on Windows with X11 support



- Use **MobaXterm** to Run Graphical Linux App
  - [tabbed] Sessions management
  - X11 server w. enhanced X extensions
  - SSH gateway / tunnels wizards
  - **Run your WSL Ubuntu profile by default**
    - ✓ share Ubuntu home (and SSH setup)

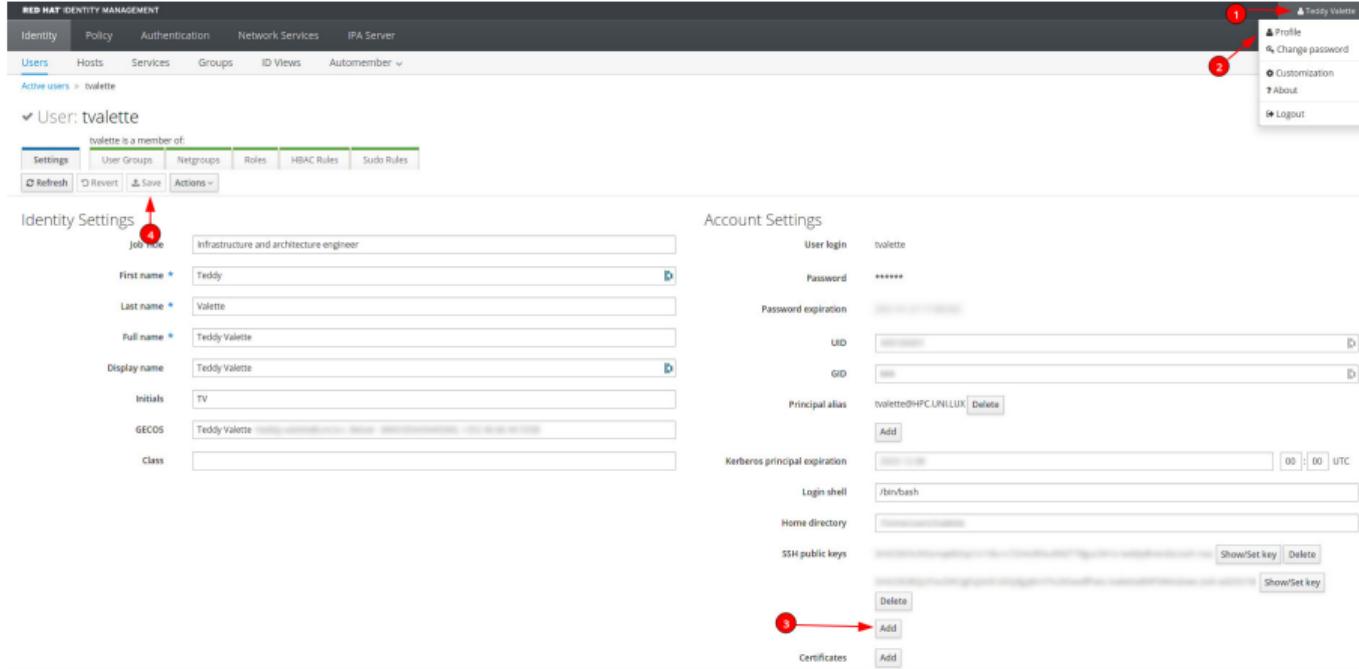


## Send your public key to ULHPC



- Connect to **ULHPC IPA portal** (the url is in your registration email)
  - ↪ Go to **your profile** page (step 1 & 2)
  - ↪ Add your **PUBLIC** SSH key (step 3)
  - ↪ **Save** your profile (step 4)

# Send your public key to ULHPC



The screenshot shows the Red Hat Identity Management interface for managing user profiles. The top navigation bar includes tabs for Identity, Policy, Authentication, Network Services, and IPA Server. The main menu has options for Users, Hosts, Services, Groups, ID Views, Automember, and Active users (with a selection for 'tvallette'). The current view is on the 'Users' tab, showing the details for the user 'tvallette'. The user's first name is 'Teddy', last name is 'Valette', full name is 'Teddy Valette', display name is 'Teddy Valette', initial is 'TV', GECOS is 'Teddy Valette', and class is empty. Under 'Identity Settings', the job role is listed as 'Infrastructure and architecture engineer'. On the right side, the 'Account Settings' panel displays the user login as 'tvallette', password as '\*\*\*\*\*', and various system identifiers like UID, GID, and Principal alias. In the bottom right corner of the account settings, there is a section for 'SSH public keys' with buttons for 'Show/Set key' and 'Delete'. A red arrow labeled '3' points to the 'Add' button under 'Certificates'. At the top right, a dropdown menu for the user 'tvallette' is open, showing options: Profile (highlighted with a red circle labeled '1'), Change password, Customization, About, and Logout. A red arrow labeled '2' points to the 'Profile' option in the dropdown.

## SSH in Practice

~/.ssh/config

```
$> ssh [-X] [-p <port>] <login>@<hostname>  
# Example: ssh -p 8022 svarrette@access-iris.uni.lu
```

```
Host <shortname>  
  Port <port>  
  User <login>  
  Hostname <hostname>
```

- ~/.ssh/config:
  - ↪ Simpler commands
  - ↪ Bash completion \$> ssh iri<TAB>

## SSH in Practice

~/.ssh/config

```
$> ssh [-X] [-p <port>] <login>@<hostname>  
# Example: ssh -p 8022 svarrette@access-iris.uni.lu
```

```
Host *.ext_ul  
    ProxyCommand ssh -q iris-cluster \  
        -W `basename %h .ext_ul`:%p  
# UL HPC Platform -- http://hpc.uni.lu  
Host iris-cluster  
    Hostname access-iris.uni.lu  
Host aion-cluster  
    Hostname access-aion.uni.lu  
Host *-cluster  
    User login #ADAPT accordingly  
    Port 8022  
    ForwardAgent no
```

```
Host <shortname>  
    Port <port>  
    User <login>  
    Hostname <hostname>
```

- ~/.ssh/config:
  - ↳ Simpler commands
  - ↳ Bash completion \$> ssh iri<TAB>

# SSH in Practice

~/.ssh/config

```
$> ssh [-X] [-p <port>] <login>@<hostname>  
# Example: ssh -p 8022 svarrette@access-iris.uni.lu
```

```
Host *.ext_ul  
    ProxyCommand ssh -q iris-cluster \  
        -W `basename %h .ext_ul`:%p  
# UL HPC Platform -- http://hpc.uni.lu  
Host iris-cluster  
    Hostname access-iris.uni.lu  
Host aion-cluster  
    Hostname access-aion.uni.lu  
Host *-cluster  
    User login #ADAPT accordingly  
    Port 8022  
    ForwardAgent no
```

```
Host <shortname>  
    Port <port>  
    User <login>  
    Hostname <hostname>
```

- ~/.ssh/config:
  - ↳ Simpler commands
  - ↳ Bash completion \$> ssh iri<TAB>

```
$> ssh iris-cluster  
$> ssh work  
$> ssh work.ext_ul
```

# SSH Training - Application to Uni.lu HPC Access

Your Turn!

## Hands-on SSH

▶ url ◀ | [github](#) | [src](#)

- **SSH Key Management**

↪ Generate strong RSA (4096 bits) & [reference] ED25519 Key pairs

- **Upload the keys on the ULHPC Identity Management Portal**

- **Custom SSH Configuration**

`~/.ssh/config`

↪ (advanced users only) SSH Agent, SOCKS Proxy

# SSH Training - Application to Uni.lu HPC Access

Your Turn!

## Hands-on SSH

▶ url ◀ | [github](#) | [src](#)

- **SSH Key Management**

↪ Generate strong RSA (4096 bits) & [reference] ED25519 Key pairs

- **Upload the keys on the ULHPC Identity Management Portal**

- **Custom SSH Configuration**

`~/.ssh/config`

↪ (advanced users only) SSH Agent, SOCKS Proxy

## Hands-on Access to Uni.lu HPC Facility

▶ url ◀ | [github](#) | [src](#)

- **Connect** to UL HPC (Linux / Mac OS / Unix)

(Step 1)

## Data transfer using SSH - SCP

- SCP (Secure CoPy):
  - ↪ from your local terminal
  - ↪ plain linear copy

```
$> scp [-P <port>] [remote:]src/ [remote:]dst/
```

```
# If you have configured your SSH client as explained before
scp /tmp/test_file iris-cluster:~
# If you haven't
scp -P 8022 /tmp/test_file tvalette@access-iris.uni.lu:~/
```

## Data transfer using SSH - RSync

- RSync (Remote SYNChronization):
  - ↪ from your local terminal
  - ↪ algorithm to optimize the transfer of files
    - ✓ based on differences between local files and remote ones

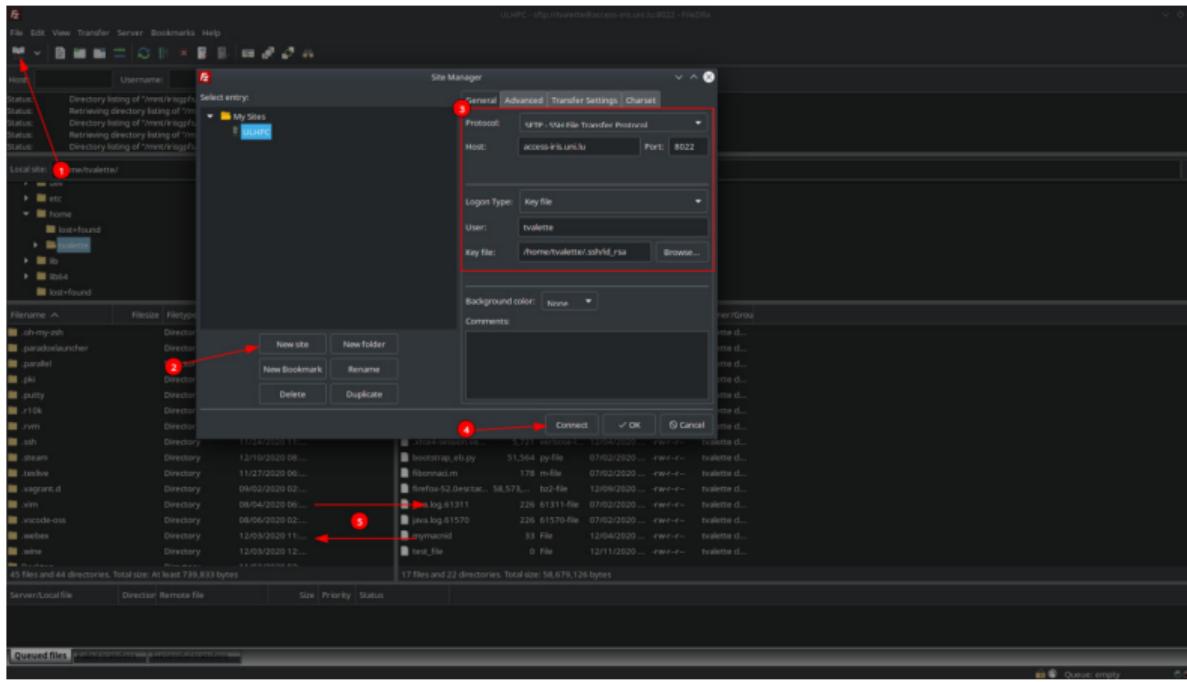
```
$> rsync [-e '<ssh options>'] [remote:]src/ [remote:]dst/
```

```
# If you have configured your SSH client as explained before
rsync /tmp/test_file iris-cluster:~
# If you haven't
rsync -e 'ssh -p 8022' /tmp/test_file tvallette@access-iris.uni.lu:~/
```

## Data transfer using SSH - FileZilla

- FileZilla:
  - Well known GUI (with a big community) compatible with Windows, Linux and MacOS
  - Open "Site Manager" (step 1)
  - Create a new site (step 2)
  - Fill the following fields (step 3)
    - ✓ Protocol: SFTP - SSH File Transfer Protocol
    - ✓ Host: access-iris.uni.lu
    - ✓ Port: 8022
    - ✓ Logon Type: Key file
    - ✓ User:
    - ✓ Key file:
  - Connect and drag and drop files from your local to your remote site and vice versa

# Data transfer using SSH - FileZilla



# Data transfer Training

Your Turn!

## Hands-on Data transfer using SSH

▶ url ◀ | [github](#) | [src](#)

- **Install your favorite tool**

- ↪ Mac OS/Linux : scp, rsync, Filezilla
- ↪ Windows : WSL (scp, rsync), Filezilla
- ✓ You can also use the MobaXTerm's data transfer feature

- Start transferring your files

wget, scp, rsync, MobaXterm(SFTP)

## ULHPC School Tutorials

- You **need to use git to get training materials** required required for Practical Sessions
  - ↪ Training materials are stored on Github: [ULHPC/tutorials](#) repository
  - ↪ You should **clone** this repository:
    - ✓ **locally** on your laptop to have quick access to the material offline
    - ✓ **remotely** within your ULHPC homedir to be able to practice and follow the PS
  - ↪ At any point of time, you can **pull** the latest updates      `git pull (or better) make up)`

- **Pre-requisite:** make sure you have a working SSH and git environment!

## ULHPC School Tutorials

- You **need to use git to get training materials** required required for Practical Sessions
  - ↪ Training materials are stored on Github: [ULHPC/tutorials](#) repository
  - ↪ You should **clone** this repository:
    - ✓ **locally** on your laptop to have quick access to the material offline
    - ✓ **remotely** within your ULHPC homedir to be able to practice and follow the PS
  - ↪ At any point of time, you can **pull** the latest updates      `git pull (or better) make up)`

- **Pre-requisite:** make sure you have a working SSH and git environment!

Your Turn!

- Open your terminal (iTerm2, Terminator, MobaXTerm, ...) & follow next instructions

## Keep Sync with Uni.lu HPC Training Material

```
### First time: clone the repo under a meaningful path
$> mkdir -p ~/git/github.com/ULHPC
$> cd ~/git/github.com/ULHPC
$> git clone https://github.com/ULHPC/tutorials.git
$> cd tutorials
$> make setup
### Next times: pull latest changes
$> cd ~/git/github.com/ULHPC/tutorials
$> make up      # update both branches (production and devel)

# Prepare a dedicated (separated) working directory
$> mkdir -p ~/tutorials/ULHPC-School-2020
$> cd ~/tutorials/ULHPC-School-2020
# create a symbolic link pointing to the tutorial reference material
$> ln -s ~/git/github.com/ULHPC/tutorials ref.d
# Now $HOME/tutorials/ULHPC-School-2020/ref.d/ points to reference training material
```

# Summary

## 1 Introduction

## 2 Preliminaries Setup on your Laptop/Workstation

System Environment Adaptation

Distributed Version Control with Git

## 3 SSH Secure Shell

Installation

Usage

Data transfer using SSH

Prepare your laptop for next Practical Sessions

## 4 Discover Open OnDemand

## ULHPC Open OnDemand (OOD) Portal



- **Open OnDemand (OOD)**

- Web portal
  - ✓ compatible with Windows, Linux and MacOS (thanks to browsers)
  - ✓ use your ULHPC credential to login
- Provides a web access to the HPC resources and integrates
  - ✓ a file management system
  - ✓ a job management system (job composer, monitoring your submitted jobs, ...)
  - ✓ an interactive command-line shell access
  - ✓ interactive apps with graphical desktop environments

## ULHPC Open OnDemand (OOD) Portal



- **Open OnDemand (OOD)**

- Web portal
  - ✓ compatible with Windows, Linux and MacOS (thanks to browsers)
  - ✓ use your ULHPC credential to login
- Provides a web access to the HPC resources and integrates
  - ✓ a file management system
  - ✓ a job management system (job composer, monitoring your submitted jobs, ...)
  - ✓ an interactive command-line shell access
  - ✓ interactive apps with graphical desktop environments

### Warning!

- The **ULHPC OOD portal** is **NOT** accessible outside the UniLu network
  - If you want to use it, you will need to setup a VPN to access the **UniLu network** [vpn.uni.lu](http://vpn.uni.lu)

## ULHPC Open OnDemand (OOD) Portal



- **Open OnDemand (OOD)**

- Web portal
  - ✓ compatible with Windows, Linux and MacOS (thanks to browsers)
  - ✓ use your ULHPC credential to login
- Provides a web access to the HPC resources and integrates
  - ✓ a file management system
  - ✓ a job management system (job composer, monitoring your submitted jobs, ...)
  - ✓ an interactive command-line shell access
  - ✓ interactive apps with graphical desktop environments

### Warning!

- The **ULHPC OOD portal** is **NOT** accessible outside the UniLu network
  - If you want to use it, you will need to setup a VPN to access the **UniLu network** [vpn.uni.lu](http://vpn.uni.lu)
  - The portal is in (*still*) under active development state.
    - ✓ Missing features and bugs can be reported to the ULHPC team via the **support portal**

# Open OnDemand - File management



The screenshot shows a file management interface with the following details:

Toolbar buttons: View, Edit, Rename/Move, Download, Copy, Paste, (Un)Select All, Delete, Go To..., Open In Terminal, New File, New Dir, Upload, Show Dotfiles, Show Owner/Mode.

Path: /home/users/tvallette/

File List:

name	size	modified date
Desktop	dir	12/04/2020
ondemand	dir	12/04/2020
bootstrap_eb.py	50.36kb	07/02/2020
fibonacci.m	178b	07/02/2020
firefox-52.0esr.tar.bz2	55.86mb	12/09/2020
java.log.61311	226b	07/02/2020
java.log.61570	226b	07/02/2020
mymachid	33b	12/04/2020
test_file	0b	12/11/2020

# Open OnDemand - Job composer

## Jobs

+ New Job ▾

★ Create Template

Show 25 entries

Search:

Created	Name	ID	Cluster	Status
December 11, 2020 4:27pm	(default) Simple Slurm Job		Iris	Not Submitted

Showing 1 to 1 of 1 entries

Previous  Next

Job Details

Job Name:  
**(default) Simple Slurm Job**

Submit to:

Account:  
Not specified

Script location:

Script name:

Folder Contents:



UNIVERSITÉ DU LUXEMBOURG

# Open OnDemand - Jobs list

Your Jobs • All Clusters •

### Active Jobs

Show 50 entries Filter:

ID	Name	User	Account	Time Used	Queue	Status	Cluster	Actions
▼ 2174794	sbatch	tvallette	uhpc	00:00:00	batch	<span style="background-color: green; color: white;">Completed</span>	Iris	<a href="#">View</a>

Completed sbatch  
2174794

Cluster	Iris
Job Id	2174794
Job Name	sbatch
User	tvallette
Account	uhpc
Partition	batch
State	COMPLETED
Reason	None
Total Nodes	1
Node List	iris-077
Total CPUs	1
Time Limit	1:00
Time Used	0:00
Memory	4096M

Output Location:

[Open in File Manager](#) [Open in Terminal](#) [Delete](#)

## Open OnDemand - Shell access

```
=====
          _/\_ 
         / \ \_/\_ \_/\_ \_/\_ 
        / \_ \_/\_ \_/\_ \_/\_ 
       / \_ \_/\_ \_/\_ \_/\_ 
      / \_ \_/\_ \_/\_ \_/\_ 
     / \_ \_/\_ \_/\_ \_/\_ 
    / \_ \_/\_ \_/\_ \_/\_ 
   / \_ \_/\_ \_/\_ \_/\_ 
  / \_ \_/\_ \_/\_ \_/\_ 
 / \_ \_/\_ \_/\_ \_/\_ 
/ \_ \_/\_ \_/\_ \_/\_ 
=====

*** Computing Nodes *****
#RAM/n *** #Cores ***
iris-[001-108] 108 Dell C6320 (2 Xeon E5-2680v4@2.4GHz [14c/128M]) 128GB 3024
iris-[109-168] 68 Dell E6420 (2 Xeon Gold 6132@2.6GHz [14c/148M]) 128GB 1688
iris-[169-186] 18 Dell C4140 (2 Xeon Gold 6132@2.6GHz [14c/148M]) 768GB 584
           +72 GPU (4 Tesla V100 [5120c CUDA + 640c Tensor]) 16GB +368640
iris-[187-198] 4 Dell R840 (4 Xeon Platin 8180@2.5GHz [28c/205W]) 3TB 448
iris-[191-196] 6 Dell C4140 (2 Xeon Gold 6132@2.6GHz [14c/148M]) 768GB 168
           +24 GPU (4 Tesla V100 [5120c CUDA + 640c Tensor]) 32GB +122880
=====
*** TOTAL: 196 nodes, 5824 cores + 491520 CUDA cores + 61440 Tensor cores ***
Fast interconnect using InfiniBand EDR 100 Gb/s technology
Shared Storage (raw capacity): 2188 TB (GPFS) + 1988 TB (Lustre) + 3480 TB

Support (in this order!):
- User DOC ..... https://hpc.uni.lu/docs
- FAQ ..... https://hpc.uni.lu/faq
- Mailing-list .... hpc-users@uni.lu
- Bug reports .NEW. https://hpc.uni.lu/support (Service Now)
- Admins ..... hpc-team@uni.lu (OPEN TICKETS)

ULHPC user guide 2020 available on hpc.uni.lu:
https://hpc.uni.lu/blog/2020/ulhpc-user-guide-2020/
=====
// NEVER COMPILE OR RUN YOUR PROGRAMS FROM THIS FRONTEND !
First reserve your nodes (using srun/sbatch[1])
linux access@iris-cluster.uni.lu:3.10.0-1062.1.1.el7.x86_64 x86_64
16:19:35 up 49 days, 5:59, 55 users, load average: 1.05, 1.37, 2.15
[16:19:35] tvallette@access[iris-cluster] ~>
```

## Open OnDemand - Interactive sessions

Iris Desktop (advanced allocation) (2174763) Queued

**Created at:** 2020-12-11 16:19:54 CET 

**Time Requested:** 1 hour

**Session ID:** [ce162143-0ec0-466b-9e49-7ac8e30cba1a](#)

Please be patient as your job currently sits in queue. The wait time depends on the number of cores as well as time requested.

Iris Desktop (advanced allocation) (2174763) 1 node | 1 core | Running

**Host:** [Iris-067](#) 

**Created at:** 2020-12-11 16:19:54 CET

**Time Remaining:** 59 minutes

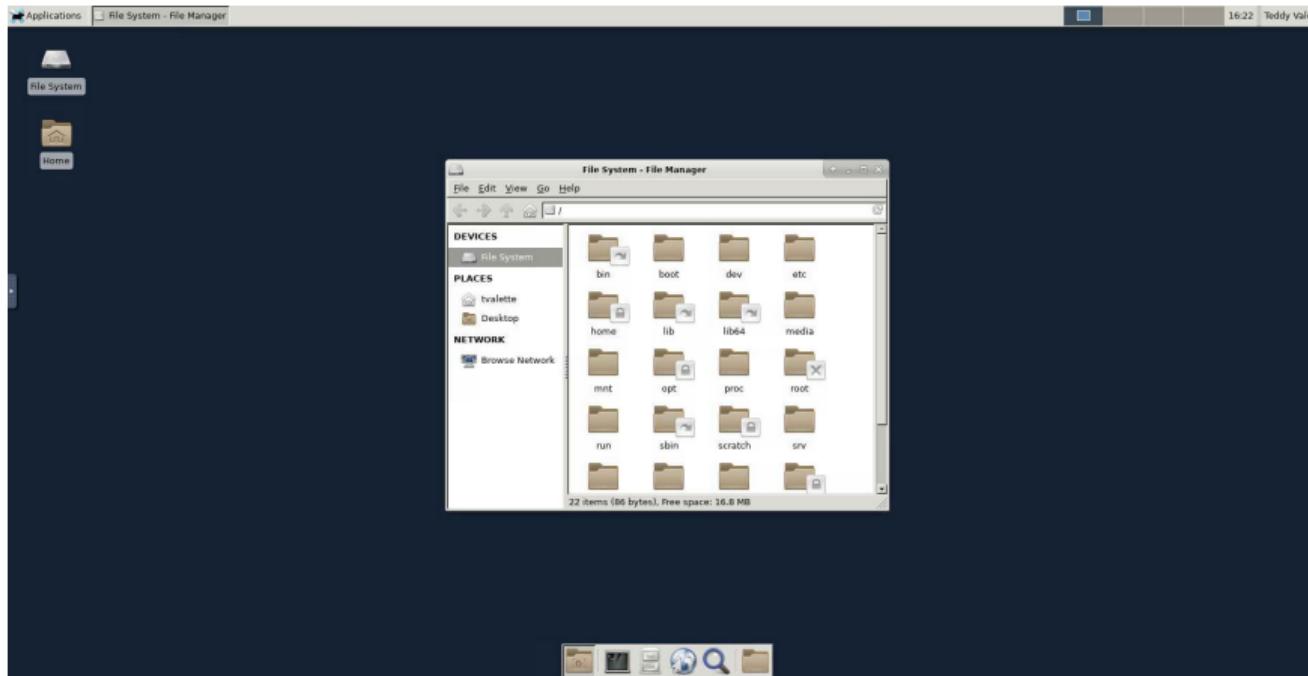
**Session ID:** [ce162143-0ec0-466b-9e49-7ac8e30cba1a](#)

**Compression** Image Quality

0 (low) to 9 (high) 0 (low) to 9 (high)

**Launch Iris Desktop (advanced allocation)** **View Only (Share-able Link)**

# Open OnDemand - Graphical Desktop Environment

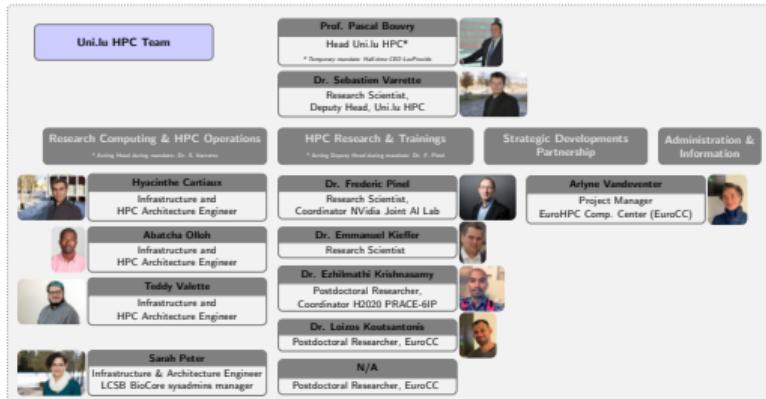


Thank you for your attention...



# Questions?

## High Performance Computing @ Uni.lu



The chart illustrates the structure of the Uni.lu HPC Team. At the top left is the "Uni.lu HPC Team". Below it, under "Research Computing & HPC Operations", are profiles for Hyacinthe Cartiaux, Ahatcha Olloh, Teddy Valette, and Sarah Peter. Under "HPC Research & Trainings", are profiles for Dr. Frederic Pimed, Dr. Emmanuel Kieller, Dr. Ezhilmarthi Krishnamoorthy, and Dr. Loïnos Koutsantonis. Under "Strategic Developments Partnership", is a profile for Ariyne Vandevenne. Under "Administration & Information", there is a "N/A" entry. At the top center is a profile for Prof. Stephane Pallage, Rector. To his right is a profile for Prof. Pascal Boutry, Head Uni.lu HPC\*, with a note indicating he is a temporary member. Below these are profiles for Dr. Sébastien Varette, Research Scientist, Deputy Head, Uni.lu HPC, and Dr. Frédéric Pimed, Research Scientist, Coordinator NVIDIA Joint AI Lab.

University of Luxembourg, Belval Campus  
Maison du Nombre, 4th floor  
2, avenue de l'Université  
L-4365 Esch-sur-Alzette  
mail: [hpc@uni.lu](mailto:hpc@uni.lu)

### 1 Introduction

### 2 Preliminaries Setup on your Laptop/Workstation

System Environment Adaptation  
Distributed Version Control with Git

### 3 SSH Secure Shell

Installation  
Usage  
Data transfer using SSH  
Prepare your laptop for next Practical Sessions

### 4 Discover Open OnDemand

<https://hpc.uni.lu>