

HELIOS

About

- HELIOS is a computer cluster
 - computers are aggregated together to create a more powerful machine.
 - User can access a specific machine (node) inside the cluster
- or
- use multiple nodes at the same time for task that are computation heavy



Connect to Helios

- Should have received your logging info
- If not ask for it to Mathieu Germain

Connect with linux

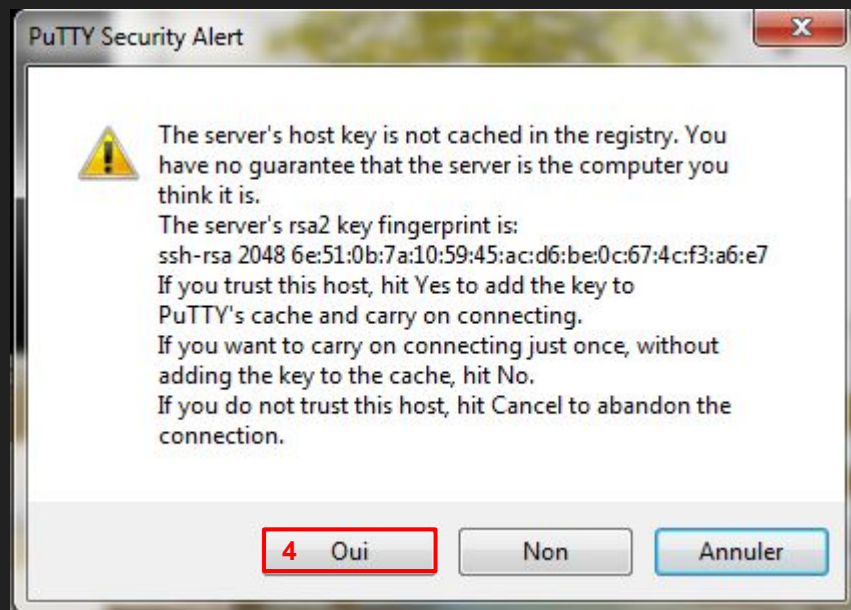
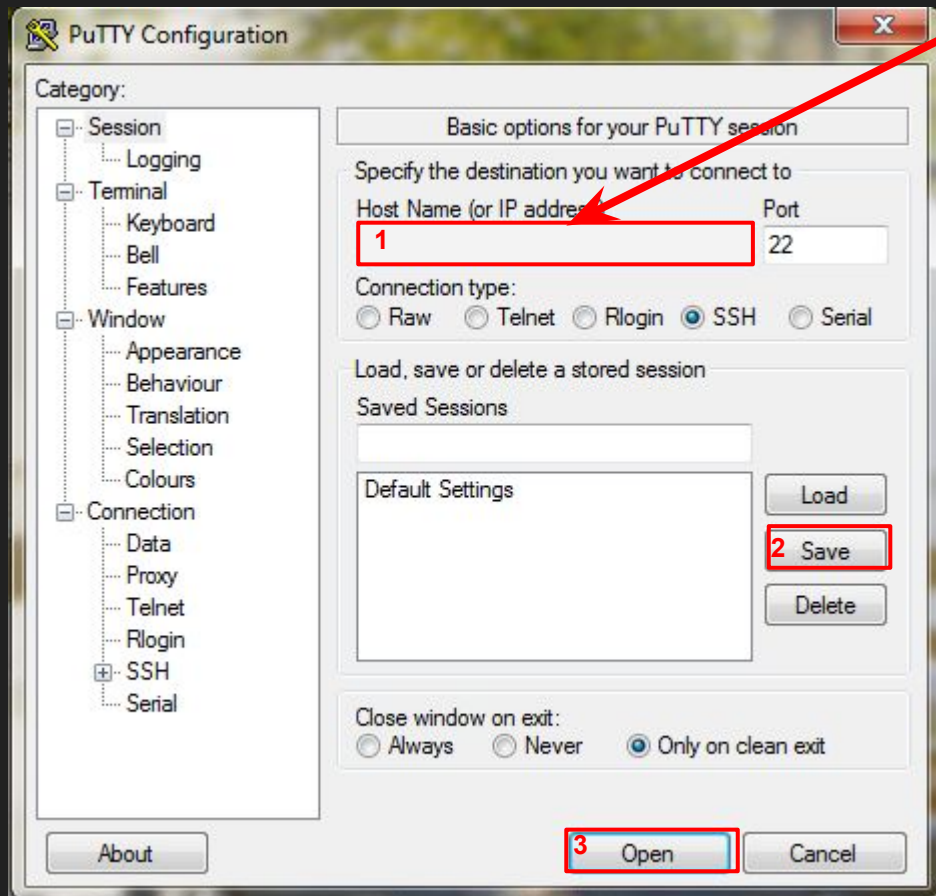
- `ssh <account-name>@helios.calculquebec.ca`

Connect with windows

- Multiple solutions
 - a. Windows subsystem for linux
 - Makes ubuntu command line available on your windows machine
 - Can install all ubuntu packages
 - b. Install putty
 - This is a ssh client for windows
 - c. Use git bash
 - Git comes with ssh

Putty

<account-name>@helios.calculquebec.ca



HELIOS

- Once connected you should see something like this →

```
user1@mila-145D ~ /helios 1 master  ssh user60@helios.calculquebec.ca
user60@helios.calculquebec.ca's password:
Last login: Wed Jan  9 10:52:58 2019 from 69.28.236.4
=====
Vous êtes sur un noeud de login de Helios (Calcul Québec).
- Nous n'effectuons pas de sauvegarde de vos fichiers.
- N'utilisez pas le noeud de login pour executer votre code.

This is a Calcul Québec login node for Helios.
- There is no backup of users files.
- Do not use this node to run code.

Rapportez tout problème à / Report any problems to: helios@calculquebec.ca
Documentation: https://wiki.calculquebec.ca/
Suivre sur Twitter/Follow on Twitter: https://twitter.com/CQ_Helios
État des serveurs: http://serveurscq.computecanada.ca
=====
Vous pouvez maintenant utiliser l'environnement logiciel de Calcul Canada
en utilisant la commande

source /admin/bin/enable_cc_cvmfs

ou

touch $HOME/.helios_ccstack

(cette dernière sera persistante)
=====
You can now use the software environment of Compute Canada by running
the command

source /admin/bin/enable_cc_cvmfs

or

touch $HOME/.helios_ccstack

(this last one will be persistent)
=====
Due to MODULEPATH changes, the following have been reloaded:
  1) openmpi/2.1.1

[user60@helios1 ~]$
```

Bash - A few useful functions

- **mkdir:** make directory

```
[delaunay@helios1 ~]$ mkdir my_project
```

- **ls:** list directories

```
[delaunay@helios1 ~]$ ls  
my_project  script_moab
```

- **cd:** change directory

```
[delaunay@helios1 ~]$ cd my_project/  
[delaunay@helios1 my_project]$
```

- **pwd:** print working directory

```
[delaunay@helios1 my_project]$ pwd  
/home/delaunay/my_project
```

- **cat:** show file content

```
[delaunay@helios1 ~]$ cat .bashrc
```

- **vi:** open a file with vi
 - to edit file press i
 - to quit press ESC then enter :q

```
[delaunay@helios1 ~]$ vi .bashrc
```


How to run AI Stuff

- Compute Resource are shared among a lot of people
- You need to demand access to those resources
- The resources will be allocated to you by the ...
 - resource manager (torque) / job scheduler: moab
 - <https://wiki.calculquebec.ca/w/Moab/en>

How to run AI Stuff

```
> msub hello.pbs
```

- Submit a job
- `hello.pbs` is a bash script

```
#!/bin/bash
#PBS -A colosse-users
#PBS -l advres=MILA2019
#PBS -l feature=k80
#PBS -l nodes=1:gpus=1
#PBS -l walltime=XX:XX:XX
```

```
echo "hello world"
```

- Small jobs have priority!
- max **12h** of runtime

How to run AI Stuff

```
[delaunay@helios1 MixedPrecisionTutorial]$ msub hello.pbs
```

371993

Job ID



```
[delaunay@helios1 MixedPrecisionTutorial]$ ls -all | grep 371993
-rw----- 1 delaunay jvb-000-01 1406 Dec 13 14:19 371993.err
-rw----- 1 delaunay jvb-000-01   64 Dec 13 14:19 371993.out
```

Job Output



Demo

```
# copy the example locally
git clone https://github.com/Delaunay/helios

# enter the example
cd helios

# Schedule the example to be run
msub hello.pbs
showq -w user=user60

# Show <job_id>.out
watch tail -n 20 $(ls -rt | grep .out | tail -n 1)
```

How to run AI Stuff

- Singularity
 - Sandboxed Execution
 - You have control over the container (image)
 - Pre configured container to simplify your life

```
singularity exec --nv --bind source:dest container script.sh
```

- `exec` : execute a script
- `--nv` : mount NVIDIA GPUs
- `--bind` : make host folder (source) available inside the container (dest)
- `container:` image of the container you want to use
- `script.sh:` the script you want to run

How to run AI Stuff

```
#!/bin/bash
#PBS -A colosse-users
#PBS -l advres=MILA2019
#PBS -l feature=k80
#PBS -l nodes=1:gpu=1
#PBS -l walltime=XX:XX:XX

module --force purge
PATH=$PATH:/opt/software/singularity-3.0/bin/

# set the working directory to where the job is launched
cd "${PBS_O_WORKDIR}"

# Singularity options
IMAGE=/rap/jvb-000-aa/COURS2019/etudiants/ift6759.simg

RAP=/rap/jvb-000-aa/COURS2019/etudiants/$USER
mkdir -p $RAP
FOLDERS=$RAP,$HOME

SINGULARITY_EXEC="singularity exec --nv --bind $FOLDERS $IMAGE"

# start your python script
$SINGULARITY_EXEC python mnist.py
```

How to run AI Stuff

```
# Schedule the example to be run
msub singularity.pbs

# Show <job_id>.out
watch tail -n 20 $(ls -rt | grep .out | tail -n 1)
```

How to run AI stuff interactively

```
> msub -N debug -A colosse-users -l advres=MILA2019,feature=k80,nodes=1:gpus=1,walltime=15:00 -I -qtest  
> module --force purge  
> PATH=$PATH:/opt/software/singularity-3.0/bin/  
> singularity shell --nv --bind $RAP,$HOME /rap/jvb-000-aa/COURS2019/etudiants/ift6759.simg
```

- Useful for testing & debugging
- The walltime/user time allocated to them is small

Monitor Jobs

- Email notification from moab
- `showq -w user=<username>`
 - Show the current jobs running for you
- `checkjob <jobid>`
 - Show details on a particular job (resource usage, status)
- `mjobctl -c <jobid>`
 - Cancel job

Quality of life

- Modify your ~/.bashrc to pre configure your environment

```
vi ~/.bashrc
```

- Add at the end of the file:

```
source /rap/jvb-000-aa/COURS2019/etudiants/common.env
```

- This will configure your environment and provide a few shortcuts

Quality of Life

- Shortcuts:
 - `mdebug`
 - Start an interactive session
 - `s_shell`
 - Start a singularity shell
 - `s_exec`
 - Execute a command inside singularity
 - `show_out`
 - monitor latest output log
 - `show_err`
 - monitor latest error log

Monitor resource usage

- RAM and CPU usage monitoring

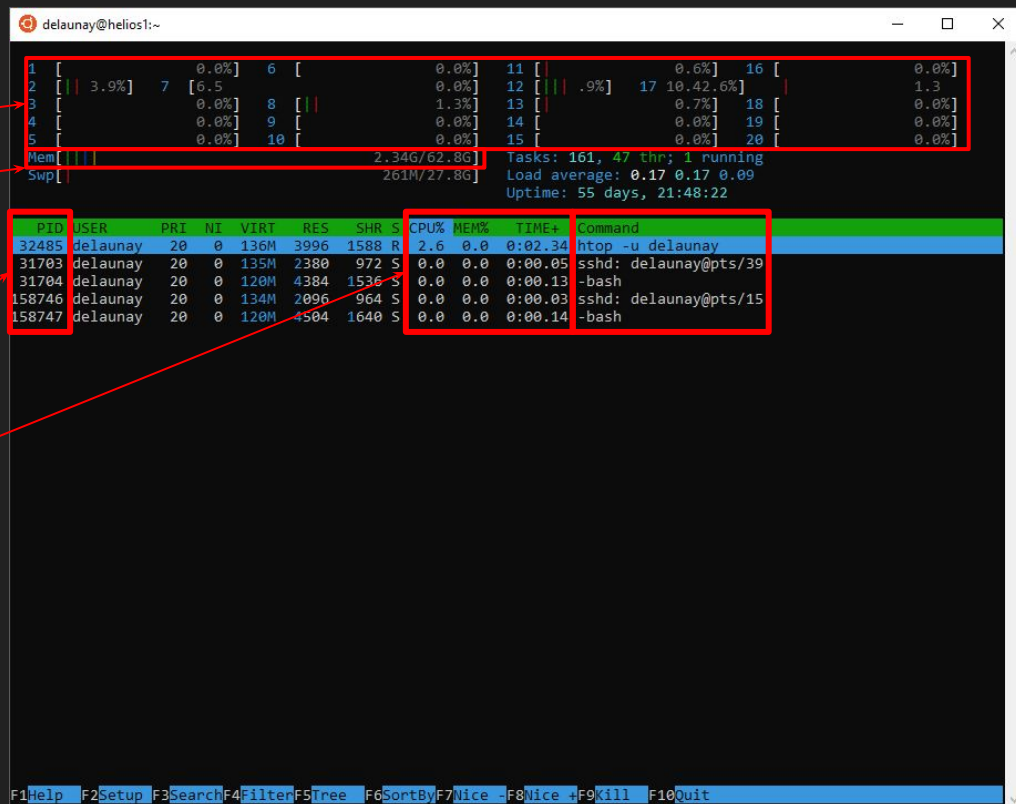
- `htop -u <username>`

- % usage per cores
 - Memory usage

- PID: Program Identifier
 - Resource usage per program

- To kill a running program:

- `kill -9 <PID>`



Monitor resource usage

- GPU usage monitoring
 - nvidia-smi

```
[delaunay@gpu-k20-03 ~]$ nvidia-smi
Thu Dec 13 12:35:39 2018
```

NVIDIA-SMI 410.73		Driver Version: 410.73		CUDA Version: 10.0	
GPU	Name	Persistence-M	Bus-Id	Disp.A	Volatile Uncorr. ECC
Fan	Temp	Perf	Pwr:Usage/Cap	Memory-Usage	GPU-Util Compute M.
=====					
0	Tesla K20m	On	00000000:05:00.0	Off	0
N/A	23C	P8	16W / 225W	0MiB / 4743MiB	0% E. Process

Processes:					GPU Memory
GPU	PID	Type	Process name		Usage
=====					
No running processes found					

Software versions

GPU Compute Usage

GPU Memory Usage

Program using GPU

Monitor resource usage

- `nvidia-smi --loop=1 --query-gpu=utilization.gpu,utilization.memory,memory.used,memory.total --id=0 --format=csv`

```
[delaunay@gpu-k20-03 ~]$ nvidia-smi --loop=1 --query-gpu=utilization.gpu,utilization.memory,memory.used,memory.total --id=0 --format=csv
utilization.gpu [%], utilization.memory [%], memory.used [MiB], memory.total [MiB]
0 %, 0 %, 0 MiB, 4743 MiB
0 %, 0 %, 0 MiB, 4743 MiB
0 %, 0 %, 0 MiB, 4743 MiB
0 %, 0 %, 0 MiB, 4743 MiB
0 %, 0 %, 0 MiB, 4743 MiB
0 %, 0 %, 0 MiB, 4743 MiB
0 %, 0 %, 0 MiB, 4743 MiB
0 %, 0 %, 0 MiB, 4743 MiB
```

- `--loop=1` run nvidia-smi every second
- `--query-gpu` specify which statistic to print
- `--id=0` show statistic only for the first gpu
- `--format=csv` print each iteration as a new CSV line

More documentation

- How to use Compute Canada Clusters
 - <https://github.com/SMART-Lab/smartdispatch/wiki/How-To-Use-Compute-Canada-Clusters>
 - <https://docs.computecanada.ca/wiki/Python>
- nvidia-smi
 - <http://developer.download.nvidia.com/compute/DCGM/docs/nvidia-smi-367.38.pdf>
- Singularity
 - <https://www.sylabs.io/guides/3.0/user-guide/index.html>