**Myriad-HPC notes ver 0.3**

Restuadi

# **Accessing myriad-HPC**

Connect to myriad-HPC using ssh protocols:

$ ssh [USERNAME@myriad.rc.ucl.ac.uk](mailto:USERNAME@myriad.rc.ucl.ac.uk)

But, most of the time, it is good to directly connect to the node you usually use for your work. For example, I did all my work in node-13, so I connected like this :

$ ssh [USERNAME@login13.myriad.rc.ucl.ac.uk](mailto:USERNAME@login13.myriad.rc.ucl.ac.uk)

This is gonna be very helpful if you get a long-time job. The first command above (without specifying the nodes) will assign you to random nodes, which are sometimes not the nodes you’re working at.

All of the connections will need a VPN to UCL networks (this includes eduroam but not UCL-guest). If you’re outside UCL but still get access to eduroam (like Imperial College), you still need a VPN.

In case the VPN fails (which is quite often), we can bypass using a direct port – which is one of the IT’s public servers (not recommended, since the IT will hate you, but sometimes you don’t have any choice if you want to work done). This is called “Tunneling” in HPC terms. Use this :

ssh -J  [USER@socrates.ucl.ac.uk](mailto:USER@socrates.ucl.ac.uk) [USER@myriad.rc.ucl.ac.uk](mailto:USER@myriad.rc.ucl.ac.uk)

Alternatively, if the -J option doesn't work, then you can run,  
  
ssh -o ProxyJump=[USER@socrates.ucl.ac.uk](mailto:USER@socrates.ucl.ac.uk) [USER@myriad.rc.ucl.ac.uk](mailto:USER@myriad.rc.ucl.ac.uk)

# **Basic operation in Myriad file system**

So there is 2 kind of file-system in myriad; in my case, it's looking like this :

[sejjr66@login13 ~]$ ls

**Scratch** standard\_RSS

You can see that the Scratch space is already mounted in our home drive.

We are supposed to only use Scratch-drive to store our inputs file and process them. Once the processing is done, we should clear up the scratch space by moving our result somewhere else (like RDS). But in practice, no one is doing this, so the RCC admin put a constrained 1 TB of scratch space for every person. This is enough for light and small sample-size operations but not for bigger computations.

In my case, I Installed my pipelines and most packages in my home. Only use the Scratch space for processing. It is safer this way since if Scratch space is wiped out due to some “unskilled HPC users” stupidity (it happened), then your installed packages and pipeline save.

The terminal has standard BASH scripting command lines.

# **Moving files HPC-RDS (Best practices)**

To access RDS, you’ll need to use a different server protocol called “transfer nodes”, so we need to connect to these transfer nodes first :

$ ssh USER@transfer02

The good thing about these transfer protocols is that all of the active RDS is already mounted, speeding up the process. For RSS RDS, it's here.

$ cd /mnt/gpfs/live/rd01\_\_/ritd-ag-project-rd010q-wlucy30/

While simple “cp” would be helpful for a couple of files, I like to use “xargs” to copy a list of files I have defined in a text file.

So the text file is formatted like this (let's say the file is called copy\_files.txt):

/mnt/gpfs/live/rd01\_\_/ritd-ag-project-rd010q-wlucy30/'BATCH 3 PR0432'/L085CD4\_S1\_R1\_001.fastq.gz

/mnt/gpfs/live/rd01\_\_/ritd-ag-project-rd010q-wlucy30/'BATCH 3 PR0432'/L085CD4\_S1\_R2\_001.fastq.gz

/mnt/gpfs/live/rd01\_\_/ritd-ag-project-rd010q-wlucy30/'BATCH 3 PR0432'/L085CD8\_S2\_R1\_001.fastq.gz

/mnt/gpfs/live/rd01\_\_/ritd-ag-project-rd010q-wlucy30/'BATCH 3 PR0432'/L085CD8\_S2\_R2\_001.fastq.gz

Then we can use :

$ xargs -a copy\_files.txt cp -t [destination\_path]

To copy the results of the pipelines, we need to copy a “Simulink” of a real binary files stored in Scratch memory, so we use a set of commands like this :

# if you haven’t connected to transfer protocols

$ ssh sejjr66@transfer02

# access the RDS folder

$ cd /mnt/gpfs/live/rd01\_\_/ritd-ag-project-rd010q-wlucy30/

# create the new folder to contain the results

$ mkdir Batch\_run\_2

$ cd Batch\_run\_2

# copy the results (for me its from Scratch) to that folder

$ cp -rL ~/Scratch/rssnextflow-v1.1/resultsUMIdevellint/ ./

# **Installing the RSS-pipelines**

1. Make sure you have connections to Myriad server etc.

2. Connect to Myriad :

You can use standard access :

$ ssh [USER@myriad.rc.ucl.ac.uk](mailto:USER@myriad.rc.ucl.ac.uk)

But I do recommend you use targeted node

$ ssh [USER@login13.myriad.rc.ucl.ac.uk](mailto:USER@login13.myriad.rc.ucl.ac.uk)

3. Open <https://gitlab.com/b8307038/rssnextflow>

or Clone directly from that github :

$ git clone <https://gitlab.com/b8307038/rssnextflow.git>

4. Copy the entire dependency for the pipelines:

$ wget -qO- https://get.nextflow.io | bash

5. Use a specific compiler :

$ module purge # is save to remove all other compilers

$ module load gcc-libs/4.9.2

6. To make sure this library installed / loaded :

$ module load compilers/gnu/4.9.2

$ module load openblas/0.2.14/gnu-4.9.2

$ module load java/1.8.0\_9 m2

7. Test the pipeline installations (inside the installation folder) :

$ ./nextflow run hello

8. Running a simple command :

$./nextflow run main.nf -with-report pipeline\_report\_RSS.html -with-timeline pipeline\_timeline\_RSS.html -with-dag pipeline.dag -c personal.config -sampleCSV samplesheet.txt

9. The main pipeline will run in the Myriad-headnode, so its good if you can use :

$ screen

Or

$ tmux

To create virtual windows that allow our pipeline to keep going after we turned-off our computer (which only serves as interface)

General HPC advice can be found here :

<https://www.rc.ucl.ac.uk/docs/>