**Setting up MQ on BMC cluster from windows client**

1. Download and install gitbash from <https://gitforwindows.org/>
2. Launch gitbash on your local windows pc and connect to cluster using **ssh -t <waftab>@<MBIOHW30.bio.med.uni-muenchen.de> 'cd </work/project/project-dir> ; bash'**

***\*stuffs inside angle brackets are variable for other systems/based on your choice and when ever I write local pc it means local windows pc***

1. Then update gcc on the Linux cluster/machine using the script from *[https://github.com/darrenjs/howto/blob/master/build\_scripts/build\_gcc\_9.sh](https://www.researchgate.net/deref/https://github.com/darrenjs/howto/blob/master/build_scripts/build_gcc_9.sh" \t "/home/wasim/Documents\\x/_blank)*

**Run the script in the terminal as:** ./build\_gcc\_9.sh

1. Download MONO from <https://download.mono-project.com/sources/mono/>
2. Install on Linux cluster using instructions from <https://www.mono-project.com/docs/compiling-mono/linux/> and add to path variable.
3. Copy MQ folder from local pc to cluster using **rsync -avP <waftab>@<MBIOHW30.bio.med.uni-muenchen.de>:</work/project/project-dir>**
4. Create MQ alias in the .bashrc file on the cluster

**Example: <MQ\_1\_6\_15\_0>=</work/project/becimh\_005/MaxQuant\_1.6.15.0/bin/MaxQuantCmd.exe>**

1. Copy raw data from local pc to cluster
2. Copy mqpar from local pc to cluster
3. Then either manually change the mqpar file for correct raw, fasta filepaths on the cluster or change them using a python script (recommended).
4. To use the script, download it from <https://github.com/wasimaftab/Utils/blob/master/gen_mqpar.py> (change the first line of the script to point to correct python path). Here I assume python is installed on the Linux machine, if not then Install it from <https://www.anaconda.com/products/individual> or any other way that you prefer.

Then to generate the mqpar file with correct path, run the script from command line as shown in the example below.

**Example:** ***./gen\_mqpar.py -in=<mqpar\_16150.xml> -fasta=</work/project/becimh\_005/timsTOF\_0720/uniprot\_P000005640\_Hsapiens\_20191126.fasta> -out=<mqpar\_test.xml> -mq=<1\_6\_15\_0> -t=<72> -raw=</work/project/becimh\_005/timsTOF\_0720>***

The above example assumes that you copied the mqpar file inside the raw data directory on Linux machine, otherwise you have to give full path to the mqpar file in the **-in** flag.

1. If you are not using Slurm in the linux machine as job scheduler then skip to step 15 else go through the steps 13-14 below.
2. After running the python script a folder named **Slurm\_Scripts** will be createdin the present working directory. Inside that folder you will find a script called **slurm.sh**

Whose entries in my case looks like below:

#!/usr/bin/sh

#SBATCH --job-name=Slurm\_Scripts

#SBATCH --output=Slurm\_Scripts.out

#SBATCH --cpus-per-task=72

#SBATCH --mem-per-cpu=256000

#SBATCH --time=2-00:00:00

#SBATCH --partition=fat

source /home/waftab/.bashrc

srun mono $MQ\_1\_6\_15\_0 /work/project/becimh\_005/timsTOF\_0720/mqpar\_test.xml

You are free to modify the parameters in this file. Pay attention to the first line, it must be modified based on bash path on your Linux machine. Also the parameters like partition, .bashrc file location etc. must be changed.

1. Now you have two choices to run MQ on Linux :
2. **Interactive mode:** Run MQ on the linux terminal using the command “srun -p <name-of-the-partition> -I -c <#-of-cpus> --mem <memory-in-MB> --time <days-hr:min:sec> mono $<MQ\_1\_6\_15\_0> <mqpar.xml>” or
3. **Submit the job:** using the script **slurm.sh** from step 13 (make sure to modify the script) as “sbatch slurm.sh”
4. In case you are not using slurm manager and/or decide to run MQ on a linux workstation rather than on a cluster then perform the steps 1-11, then simply run MQ on a Linux terminal as “mono $<MQ\_1\_6\_15\_0> <mqpar.xml>”