

## Steps to connect to clusters(Telluride/Turretarch) and run the program on the cluster

Ssh to connect to any cluster from your computer

### A. Turretarch

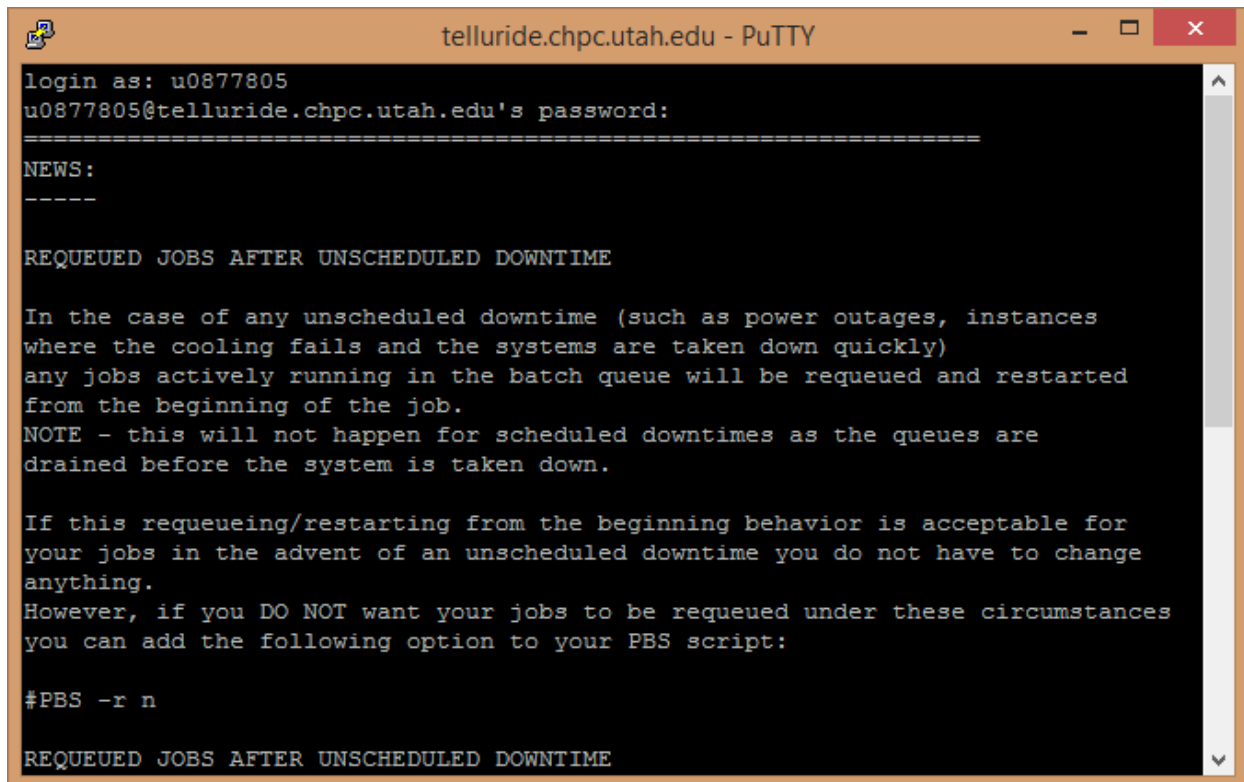
1. ssh -X turretarch10.chpc.utah.edu
2. Give your username and id to connect to the cluster

Username is UID and Password is CIS password

login as: u0877805

u0877805@telluride.chpc.utah.edu's password:

---



```
telluride.chpc.utah.edu - PuTTY
login as: u0877805
u0877805@telluride.chpc.utah.edu's password:
=====
NEWS:
-----

REQUEUED JOBS AFTER UNSCHEDULED DOWNTIME

In the case of any unscheduled downtime (such as power outages, instances
where the cooling fails and the systems are taken down quickly)
any jobs actively running in the batch queue will be requeued and restarted
from the beginning of the job.
NOTE - this will not happen for scheduled downtimes as the queues are
drained before the system is taken down.

If this requeueing/restarting from the beginning behavior is acceptable for
your jobs in the advent of an unscheduled downtime you do not have to change
anything.
However, if you DO NOT want your jobs to be requeued under these circumstances
you can add the following option to your PBS script:

#PBS -r n

REQUEUED JOBS AFTER UNSCHEDULED DOWNTIME
```

3. Touch filename to create the file on the cluster

Example:

Touch greetings.c

```
telluride.chpc.utah.edu - PuTTY
=====: Command not f ^
ound.
[u0877805@telluride2 ~]$ pwd
/uufs/chpc.utah.edu/common/home/u0877805
[u0877805@telluride2 ~]$ ls -ltra
total 96
-rw-r--r-- 1 u0877805 CS6230  9778 Jan  6 14:46 .bashrc
-rw-r--r-- 1 u0877805 CS6230 17117 Jan  6 14:46 .tcshrc
-rw-r--r-- 1 u0877805 CS6230   191 Jan  6 14:46 .bash_profile
-rw-r--r-- 1 u0877805 CS6230    24 Jan  6 14:46 .bash_logout
drwxr-xr-x 3 u0877805 CS6230   28 Jan  6 14:46 .mozilla
drwxr-xr-x 3 u0877805 CS6230   26 Jan  6 14:46 .kde
drwxr-xr-x 2 u0877805 CS6230   10 Jan  6 23:14 .pbs_spool
-rw----- 1 u0877805 CS6230 12288 Jan  8 00:51 .swp
drwxr-xr-x 3 u0877805 CS6230   24 Jan  8 00:57 intel
-rw-r--r-- 1 u0877805 CS6230  1695 Jan  8 01:00 greetings.c
-rwxr-xr-x 1 u0877805 CS6230 11148 Jan  8 01:02 greetings
-rw-r--r-- 1 u0877805 CS6230  4010 Jan  8 01:20 trap.c
-rw----- 1 u0877805 CS6230  1551 Jan  8 01:20 .viminfo
drwxr-xr-x 6 u0877805 CS6230  4096 Jan  8 01:21 .
-rwxr-xr-x 1 u0877805 CS6230 11645 Jan  8 01:21 trap
-rw----- 1 u0877805 CS6230   658 Jan  8 01:35 .history
drwxr-xr-x 7 root      root      0 Jan 16 04:51 ..
[u0877805@telluride2 ~]$
```

And then opened greetings.c using vi editor to write the program and saved it (use :wq! To save in vi editor)

Sample Program code for greetings.c

```
[u0877805@telluride2 ~]$ vi greetings.c
```

```
/* Find out number of processes */
MPI_Comm_size(MPI_COMM_WORLD, &p);

if (my_rank != 0) {
    /* Create message */
    sprintf(message, "Greetings from process %d!",
            my_rank);
    dest = 0;
    /* Use strlen+1 so that '\0' gets transmitted */
    MPI_Send(message, strlen(message)+1, MPI_CHAR,
            dest, tag, MPI_COMM_WORLD);
} else { /* my_rank == 0 */
```

```

    for (source = 1; source < p; source++) {
        MPI_Recv(message, 100, MPI_CHAR, source, tag,
                MPI_COMM_WORLD, &status);
        printf("%s\n", message);
    }
}

/* Shut down MPI */
MPI_Finalize();
} /* main */

```

4. FOR TURRETARCH edit the .bashrc

```

elif [[ "$UUFSCELL" = "turretarch.arches" ]]; then
    source /uufs/chpc.utah.edu/sys/pkg/mpich2/std/etc/mpich2.sh
# export SSIHLM=/uufs/turretarch.arches/sys/pkg/hlm/std
export SSIHLM=/uufs/chpc.utah.edu/sys/pkg/mpich2/std

```

5. To compile the greetings.c we use mpicc compiler

```

Mpicc -g -o greetings greetings.c

```

-g will is used to create information that allows us to use a debugger

-o <outfile>.Put the executable in the file named outfile

6. To run the program

```

Mpiexec -n <number of processes> ./greetings

```

or

```

mpirun -n <number of processes> ./greetings

```

## 7. OUTPUT of greetings.c

```
[u0877805@turretarch10 ~]$ mpirun -n 16 ./greetings
```

Greetings from process 1!

Greetings from process 2!

Greetings from process 3!

Greetings from process 4!

Greetings from process 5!

Greetings from process 6!

Greetings from process 7!

Greetings from process 8!

Greetings from process 9!

Greetings from process 10!

Greetings from process 11!

Greetings from process 12!

Greetings from process 13!

Greetings from process 14!

Greetings from process 15!

## B. Telluride

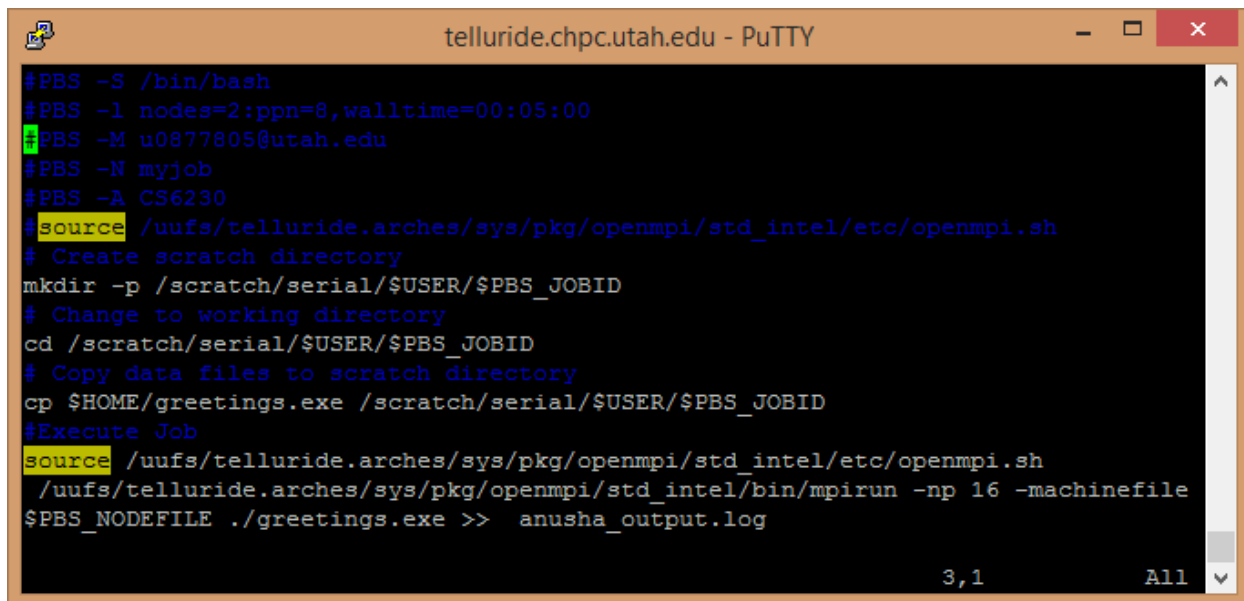
1. `ssh -X telluride.chpc.utah.edu`
2. compile the .c file

```
/uufs/telluride.arches/sys/pkg/openmpi/std_intel/bin/mpicc -o greetings  
greetings.c -I/uufs/telluride.arches/sys/pkg/openmpi/std_intel/include  
-L/uufs/telluride.arches/sys/pkg/openmpi/std_intel/lib
```

3. **FOR TELLURIDE edit the .bashrc**

```
elif [[ "$UUFSCELL" = "telluride.arches" ]]; then  
#source /uufs/telluride.arches/sys/pkg/openmpi/std_intel/etc/openmpi.sh #anusha  
source /uufs/chpc.utah.edu/sys/pkg/mkl/std/tools/environment/mklvarsem64t.sh  
source /uufs/telluride.arches/sys/pkg/openmpi/std_intel/etc/openmpi.sh  
export MPI_SHELL="/usr/bin/rsh"
```

4. Create a log file to redirect the console output to log file touch output.log
5. Touch myjob.pbs will create the batch job for telluride cluster which will look like



```
telluride.chpc.utah.edu - PuTTY  
#PBS -S /bin/bash  
#PBS -l nodes=2:ppn=8,walltime=00:05:00  
#PBS -M u0877805@utah.edu  
#PBS -N myjob  
#PBS -A CS6230  
#source /uufs/telluride.arches/sys/pkg/openmpi/std_intel/etc/openmpi.sh  
# Create scratch directory  
mkdir -p /scratch/serial/$USER/$PBS_JOBID  
# Change to working directory  
cd /scratch/serial/$USER/$PBS_JOBID  
# Copy data files to scratch directory  
cp $HOME/greetings.exe /scratch/serial/$USER/$PBS_JOBID  
#Execute Job  
source /uufs/telluride.arches/sys/pkg/openmpi/std_intel/etc/openmpi.sh  
/uufs/telluride.arches/sys/pkg/openmpi/std_intel/bin/mpirun -np 16 -machinefile  
$PBS_NODEFILE ./greetings.exe >> anusha_output.log  
3,1 All
```

Give below is the .pbs file and I have comments for each line

```
[u0877805@telluride2 ~]$ cat myjob.pbs
```

```
#PBS -S /bin/bash
```

```

#PBS -l nodes=2:ppn=8,walltime=00:05:00    ---specifies the number of processors no of nodes and
                                           processors used on each node i.e 2 nodes each 8 processors so total 16

#PBS -M u0877805@utah.edu                ---ur email id

#PBS -N myjob                               --- job name

#PBS -A CS6230                               -- to ensure that the job gets prioritized

# Create scratch directory

mkdir -p /scratch/serial/$USER/$PBS_JOBID  --creating the job_id folder in the scratch directory

# Change to working directory

cd /scratch/serial/$USER/$PBS_JOBID        --changing to scratch directory

# Copy data files to scratch directory

cp $HOME/greetings /scratch/serial/$USER/$PBS_JOBID  --copying the file to the scratch directory

#Execute Job

source /uufs/telluride.arches/sys/pkg/openmpi/std_intel/etc/openmpi.sh  --sourcing the openmpi

/uufs/telluride.arches/sys/pkg/openmpi/std_intel/bin/mpirun -np 16 -machinefile $PBS_NODEFILE
./greetings>>  output.log    ---running the file and redirecting the output to log file

```

Inside the Pbs job we will specify how many total processors you will use for your program i.e 16 in the above case and what is the file name that you would be executing

6. qsub myjob.pbs to submit the pbs job which will be put in the queue on the cluster

```
[u0877805@telluride2 ~]$ qsub myjob.pbs
```

```
47622.trrm.privatearch.arches
```

Once we submit we will check within the wait time that we have specified in the pbs file to know the job is in which state

7. Command to check whether your job is in blocked,idle,running on telluride

– showq

- -i (idle jobs)
- -r (running)
- -b (blocked)

Example of showq

```
[u0877805@telluride2 ~]$ showq -i
```

eligible jobs-----

JOBID	PRIORITY	XFACTOR	Q	USERNAME	GROUP	PROCS	WCLIMIT
CLASS	SYSTEMQUEUE	TIME					
47622	281160	1.0	-	u0877805	CS6230	16	00:15:00
telluride	Thu Jan 16 05:20:28						

1 eligible job

Total job: 1

The output will be available in your output.log file which is present in /scratch/serial/\$USER/\$PBS\_JOBID

Greetings from process 1!

Greetings from process 2!

Greetings from process 3!

Greetings from process 4!

Greetings from process 5!

Greetings from process 6!

Greetings from process 7!

Greetings from process 8!

Greetings from process 9!

Greetings from process 10!

Greetings from process 11!

Greetings from process 12!

Greetings from process 13!

Greetings from process 14!

Greetings from process 15!

NOTE :-

Two changes to .bashrc file

**FOR TURRETARCH**

```
elif [[ "$UUFSCELL" = "turretarch.arches" ]]; then
source /uufs/chpc.utah.edu/sys/pkg/mpich2/std/etc/mpich2.sh
    # export SSIHLM=/uufs/turretarch.arches/sys/pkg/hlm/std
export SSIHLM=/uufs/chpc.utah.edu/sys/pkg/mpich2/std
```

**FOR TELLURIDE**

```
elif [[ "$UUFSCELL" = "telluride.arches" ]]; then
#source /uufs/telluride.arches/sys/pkg/openmpi/std_intel/etc/openmpi.sh #anusha
    source /uufs/chpc.utah.edu/sys/pkg/mkl/std/tools/environment/mklvarsem64t.sh
source /uufs/telluride.arches/sys/pkg/openmpi/std_intel/etc/openmpi.sh
    export MPI_SHELL="/usr/bin/rsh"
```

First use Turretarch since it is used for benchmarking and then you will know the waittime and then use Telluride.