Table of Contents

- User environment
- Slurm User commands.
The cluster is accessed by a ssh connection from a client machine.

The user must use a private/public key.

The public key must be provided to the cluster administrator, so they can install it on the correct path to allow the access to the user.

```
bulladm@clientmachine$ ssh -i privatekey lattes0
```
User environment

- The users of the cluster must login on the Login node.
- They use the cluster for compilation, submit jobs to the compute nodes and monitor the job status.
- The access to compute nodes should be denied.
- They could run stand-alone or distributed jobs.
Environment for compilation

The GNU gcc/g++/gfortran compiler are installed on the login node.

- GNU gcc/gfortran v4.1.2.
  - /usr/
  - There is not need to configure the environment to use it.
  - It is in the Linux default paths.
User environment - MPI

The available implementations at the cluster are:

OpenMPI-1.5.4
BullMPI2 v1.3.9-18.s

Los compiladores son mpicc, mpif90, mpicxx, etc.
The user could set the openmpi environment on the initializations scripts (.bashrc, .bash_profile):

```
#OpenMPI
PATH=$PATH:/opt/mpi/openmpi-1.5.4/bin
LD_LIBRARY_PATH=/opt/mpi/openmpi-1.5.4/lib
export LD_LIBRARY_PATH
```
The user could set the bullmpi environment on the initializations scripts (.bashrc, .bash_profile):

```bash
#Bullmpi
source /opt/mpi/mpibull2-1.3.9-18.s/share/mpibull2.sh
mpibull2_devices -d=ibmr_gen2
export CC=gcc
export CXX=g++
export FC=gfortran
```

The ibmr_gen2 set the IB network for communications and osock set the GbE network.

The Bull mpi compilation uses the CC, CXX and FC environment variables to select the compiler.
SLURM – User commands

Ares, Gerardo, HPC Team
Architecture diagram
Training example

A sample partition definition.

PartitionName=cpu Nodes=lattes[1-15] Default=Yes
  MaxTime=500:00:00 AllowGroups=scpu,sgpu,sfatnode
  State=UP

PartitionName=gpu Nodes=lattes[16-23] Default=NO
  MaxTime=500:00:00 AllowGroups=sgpu State=UP

PartitionName=fatnodes Nodes=lattes[24-28] Default=NO
  MaxTime=500:00:00 AllowGroups=sfatnode State=UP

Users:
  - scpu (cpu only).
  - sgpu (cpu only & gpu).
  - sfatnode (scpu & fatnode).
User commands

- **sinfo**
  - Allow to see the current partition status or the current node status.

- **srun**
  - Used to submit jobs interactively.

- **sbatch**
  - Used to submit batch jobs.

- **squeue**
  - Display information about the job queue.

- **sstat**
  - Display information about the running jobs.

- **scancel**
  - Used to cancel jobs from the queue.
sinfo: display information about the partitions/nodes that the user is allowed to run

[sfatecnode@lattes0 ~]$ sinfo
PARTITION AVAIL  TIMELIMIT  NODES  STATE  NODELIST
cpu*             up   20-20:00:0      11  down  lattes[5-15]
cpu*             up   20-20:00:0      4   idle  lattes[1-4]
fatnodes         up   20-20:00:0      5   idle  lattes[24-28]

[sfatecnode@lattes0 ~]$ sinfo -N
 NODELIST        NODES PARTITION STATE
lattes[1-4]      4     cpu*   idle
lattes[5-15]     10    cpu*   down
lattes[24-28]    5     fatnodes   idle
```
[sfatnode@lattes0 serial]$ squeue

<table>
<thead>
<tr>
<th>JOBID</th>
<th>PARTITION</th>
<th>NAME</th>
<th>USER</th>
<th>ST</th>
<th>TIME</th>
<th>NODES</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>99</td>
<td>cpu</td>
<td>HPL</td>
<td>bulladm</td>
<td>PD</td>
<td>0:00</td>
<td>4 (Resources)</td>
</tr>
<tr>
<td>100</td>
<td>cpu</td>
<td>HPL</td>
<td>bulladm</td>
<td>PD</td>
<td>0:00</td>
<td>4 (Priority)</td>
</tr>
<tr>
<td>101</td>
<td>cpu</td>
<td>MPI-PI</td>
<td>scpu</td>
<td>PD</td>
<td>0:00</td>
<td>1 (Priority)</td>
</tr>
<tr>
<td>102</td>
<td>cpu</td>
<td>MPI-PI</td>
<td>scpu</td>
<td>PD</td>
<td>0:00</td>
<td>1 (Priority)</td>
</tr>
<tr>
<td>103</td>
<td>cpu</td>
<td>MPI-PI</td>
<td>scpu</td>
<td>PD</td>
<td>0:00</td>
<td>1 (Priority)</td>
</tr>
<tr>
<td>104</td>
<td>cpu</td>
<td>MPI-PI</td>
<td>scpu</td>
<td>PD</td>
<td>0:00</td>
<td>1 (Priority)</td>
</tr>
<tr>
<td>98</td>
<td>cpu</td>
<td>HPL</td>
<td>bulladm</td>
<td>R</td>
<td>1:18</td>
<td>4 lattes[1-4]</td>
</tr>
<tr>
<td>105</td>
<td>fatnodes</td>
<td>MPI-PI</td>
<td>sfatnode</td>
<td>R</td>
<td>0:03</td>
<td>1 lattes24</td>
</tr>
<tr>
<td>106</td>
<td>fatnodes</td>
<td>MPI-PI</td>
<td>sfatnode</td>
<td>R</td>
<td>0:02</td>
<td>1 lattes24</td>
</tr>
<tr>
<td>107</td>
<td>fatnodes</td>
<td>MPI-PI</td>
<td>sfatnode</td>
<td>R</td>
<td>0:01</td>
<td>1 lattes24</td>
</tr>
</tbody>
</table>
```
squeue job states

<table>
<thead>
<tr>
<th>ST</th>
<th>State</th>
</tr>
</thead>
<tbody>
<tr>
<td>PD</td>
<td>Pending</td>
</tr>
<tr>
<td>R</td>
<td>Running</td>
</tr>
<tr>
<td>CA</td>
<td>Cancelled</td>
</tr>
<tr>
<td>CG</td>
<td>Completion</td>
</tr>
<tr>
<td>CD</td>
<td>Completed</td>
</tr>
<tr>
<td>F</td>
<td>Failed</td>
</tr>
<tr>
<td>TO</td>
<td>Timeout</td>
</tr>
<tr>
<td>NF</td>
<td>Node Failure</td>
</tr>
</tbody>
</table>
scancel: cancel a schedule job.

[bulladm@lattes0 ~]$ squeue

<table>
<thead>
<tr>
<th>JOBID</th>
<th>PARTITION</th>
<th>NAME</th>
<th>USER</th>
<th>ST</th>
<th>TIME</th>
<th>NODES</th>
<th>NODELIST (REASON)</th>
</tr>
</thead>
<tbody>
<tr>
<td>99</td>
<td>cpu</td>
<td>HPL</td>
<td>bulladm</td>
<td>PD</td>
<td>0:00</td>
<td>4</td>
<td>Resources</td>
</tr>
<tr>
<td>100</td>
<td>cpu</td>
<td>HPL</td>
<td>bulladm</td>
<td>PD</td>
<td>0:00</td>
<td>4</td>
<td>Priority</td>
</tr>
<tr>
<td>101</td>
<td>cpu</td>
<td>MPI-PI</td>
<td>scpu</td>
<td>PD</td>
<td>0:00</td>
<td>1</td>
<td>Priority</td>
</tr>
<tr>
<td>102</td>
<td>cpu</td>
<td>MPI-PI</td>
<td>scpu</td>
<td>PD</td>
<td>0:00</td>
<td>1</td>
<td>Priority</td>
</tr>
<tr>
<td>98</td>
<td>cpu</td>
<td>HPL</td>
<td>bulladm</td>
<td>R</td>
<td>3:25</td>
<td>4</td>
<td>lattes[1-4]</td>
</tr>
<tr>
<td>105</td>
<td>fatnodes</td>
<td>MPI-PI</td>
<td>sfatnode</td>
<td>R</td>
<td>2:33</td>
<td>1</td>
<td>lattes24</td>
</tr>
</tbody>
</table>

[bulladm@lattes0 ~]$ scancel 98

[bulladm@lattes0 ~]$ squeue

<table>
<thead>
<tr>
<th>JOBID</th>
<th>PARTITION</th>
<th>NAME</th>
<th>USER</th>
<th>ST</th>
<th>TIME</th>
<th>NODES</th>
<th>NODELIST (REASON)</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>cpu</td>
<td>HPL</td>
<td>bulladm</td>
<td>PD</td>
<td>0:00</td>
<td>4</td>
<td>Resources</td>
</tr>
<tr>
<td>101</td>
<td>cpu</td>
<td>MPI-PI</td>
<td>scpu</td>
<td>PD</td>
<td>0:00</td>
<td>1</td>
<td>Priority</td>
</tr>
<tr>
<td>102</td>
<td>cpu</td>
<td>MPI-PI</td>
<td>scpu</td>
<td>PD</td>
<td>0:00</td>
<td>1</td>
<td>Priority</td>
</tr>
<tr>
<td>99</td>
<td>cpu</td>
<td>HPL</td>
<td>bulladm</td>
<td>R</td>
<td>0:11</td>
<td>4</td>
<td>lattes[1-4]</td>
</tr>
<tr>
<td>105</td>
<td>fatnodes</td>
<td>MPI-PI</td>
<td>sfatnode</td>
<td>R</td>
<td>2:33</td>
<td>1</td>
<td>lattes24</td>
</tr>
</tbody>
</table>
sbatch: Used to submit jobs.

$ sbatch ./pi.srm

$ cat pi.srm
#!/bin/bash
#SBATCH -J MPI-PI
#SBATCH --partition cpu
#SBATCH --nodes 4
#SBATCH --ntasks 48
#SBATCH --cpus-per-task 1
#SBATCH --time 3:00:00

cd $SLURM_SUBMIT_DIR

srun pi
sbatch parameters

- **Partition: --partition, -p**
  - Select the partition to run.

- **Job Name: --job-name, -J**
  - Name of the job.

- **Time: --time, -t**
  - Time limit for the job.

- **Begin: --begin**
  - The job will not start before that time.
  - --begin=18:00
  - --begin=now+300 (seconds)

```
[bulladm@lattes0 lattes]$ sbatch --begin=01:00 hpl-mpibull.srm
Submitted batch job 108
```

```
[bulladm@lattes0 lattes]$ squeue

<table>
<thead>
<tr>
<th>JOBID</th>
<th>PARTITION</th>
<th>NAME</th>
<th>USER</th>
<th>ST</th>
<th>TIME</th>
<th>NODES</th>
</tr>
</thead>
<tbody>
<tr>
<td>108</td>
<td>cpu</td>
<td>HPL</td>
<td>bulladm</td>
<td>PD</td>
<td>0:00</td>
<td>4</td>
</tr>
<tr>
<td>99</td>
<td>cpu</td>
<td>HPL</td>
<td>bulladm</td>
<td>R</td>
<td>2:42</td>
<td>4</td>
</tr>
</tbody>
</table>
```
sbatch resources allocation

- **Number of nodes:** `--nodes`, `-N`
  - Select the number of compute nodes.

- **Number of tasks:** `--ntasks`, `-n`
  - Number of tasks (process) for the job.

- **Cores per tasks:** `--cpus-per-task`, `-c`
  - Number of cores per tasks (multithread application).
To submit a serial process:

```bash
#SBATCH -J serial-pi
#SBATCH --partition cpu
#SBATCH --nodes 1
#SBATCH --ntasks 1
#SBATCH --cpus-per-task 1
```

- It consumes one core on a single node.
- Users could share nodes (slurm administrator).
To submit a multithread process on a single node:

```bash
#SBATCH -J MPI-PI
#SBATCH --partition cpu
#SBATCH --nodes 1
#SBATCH --ntasks 2
#SBATCH --cpus-per-task 6
```

This job needs a complete compute node to run.
To submit a MPI distributed application:

```bash
#SBATCH -J MPI-PI
#SBATCH --partition cpu
#SBATCH --nodes 4
#SBATCH --ntasks 48
#SBATCH --cpus-per-task 1
```

This job needs four complete compute node to run.
To submit a Multithread-MPI distributed application:

```
#SBATCH -J MPI-PI
#SBATCH --partition cpu
#SBATCH --nodes 8
#SBATCH --ntasks 48
#SBATCH --cpus-per-task 2
```

- This job spawn 48 process with 2 thread on each one.
- It needs 8 complete compute node to run.
To submit a Multithread-MPI distributed application:

#SBATCH -J MPI-PI
#SBATCH --partition medium
#SBATCH --nodes 8
#SBATCH --ntasks 24
#SBATCH --cpus-per-task 2

This job spawn 16 process with 2 thread on each one.
It needs 8 compute node to run, but not complete.
Provides information about the running process:

[bulladm@lattes0 serial]$ sstat -j 115

<table>
<thead>
<tr>
<th>JobID</th>
<th>MaxVMSize</th>
<th>MaxVMSizeNode</th>
<th>MaxVMSizeTask</th>
<th>AveVMSize</th>
<th>MaxRSS</th>
<th>MaxRSSNode</th>
<th>MaxRSSTask</th>
<th>AveRSS</th>
<th>MaxPages</th>
<th>MaxPagesNode</th>
<th>MaxPagesTask</th>
<th>AvePages</th>
<th>MinCPU</th>
<th>MinCPUNode</th>
<th>MinCPUTask</th>
<th>AveCPU</th>
<th>NTasks</th>
</tr>
</thead>
<tbody>
<tr>
<td>115.0</td>
<td>142660K</td>
<td>lattes24</td>
<td>0</td>
<td>142660K</td>
<td>44192K</td>
<td>lattes24</td>
<td>0</td>
<td>44192K</td>
<td>0</td>
<td>lattes24</td>
<td>0</td>
<td>lattes24</td>
<td>0</td>
<td>00:52.000</td>
<td>lattes24</td>
<td>0</td>
<td>00:52.000</td>
</tr>
</tbody>
</table>
sbatch stdout and stderr outputs

- Slurm store on the submit directory the stdout and stderr of the job.

- The file name are:
  - slurm-\$SLURM_JOBID.out (stdout)
  - slurm-\$SLURM_JOBID.err (stderr)
Environment for MPI (execution)

- The users will submit jobs by SLURM resource manager.
- For batch processing they must use sbatch command.
- For interactive processing they must use srun command.

Example of sbatch with OpenMPI:

```bash
srun -resv-ports ./mpiapplication.srm
```
SLURM – Hints

Ares, Gerardo, HPC Team
Hints

Use SLURM environment variables on the scripts.

SLURM_NODELIST=lattes[1-8]
SLURM JOB_NAME=MPI-PI
SLURM_PRIO_PROCESS=0
SLURM_NNODES=8
SLURM_JOBID=335
SLURM_TASKS_PER_NODE=12 (x12)
SLURM_JOB_ID=335
SLURM_UMASK=0022
SLURM_NODEID=0
SLURM_SUBMIT_DIR=/home_cluster/testes/test5/2011/mpi-pi
SLURM_TASK_PID=392
SLURM_NPROCS=96
SLURM_CPUS_ON_NODE=12
SLURM_JOB_NODELIST=lattes[1-8]
SLURM_JOB_CPUS_PER_NODE=12 (x12)
SLURM_JOB_NUM_NODES=8
Hints

- Print the runtime dynamic libraries environment.

```bash
#!/bin/bash
#SBATCH --J mpi-pi
date
echo "--------------------------------------"
ldd ./pi
echo "--------------------------------------"
srun --resv-ports ./pi

-----------------------------------------
libmpi.so => /opt/mpi/mpibull2-1.3.9-18.s/lib/libmpi.so (0x00002b3d92084000)
librt.so.1 => /lib64/librt.so.1 (0x0000003084200000)
libdl.so.2 => /lib64/libdl.so.2 (0x0000003083600000)
libpmi.so => /usr/lib64/libpmi.so (0x00002b3d923e1000)
libmpidev.so => /opt/mpi/mpibull2-1.3.9-18.s/lib/drivers/osock/libmpidev.so (0x00002b3d925e8000)
...
```