Artemis User Guide

Information and Communications Technology

Sydney Informatics Hub
# Table of Contents

**GETTING STARTED** .............................................................................................................. 2
  - Access to Artemis ............................................................................................................. 2
  - If you have an existing RDS account and need access to Artemis ............................... 2
  - Connecting to Artemis .................................................................................................. 2

**ARTEMIS COMPUTE NODES** .............................................................................................. 3
  - Node allocations ............................................................................................................. 3

**DATA STORAGE ON ARTEMIS** .......................................................................................... 3
  - /home ............................................................................................................................. 4
  - /project .......................................................................................................................... 4
  - /scratch .......................................................................................................................... 4

**SOFTWARE/PROGRAMS/APPLICATIONS** ......................................................................... 5

**SUBMITTING JOBS TO ARTEMIS** ..................................................................................... 6
  - Serial jobs (1 core) ........................................................................................................ 6
  - Parallel jobs using OpenMP ......................................................................................... 6
  - Parallel jobs using MPI ................................................................................................. 6
  - Job monitoring/management commands .................................................................... 7

**JOB SCHEDULING PRIORITY: “FAIR SHARE”** ............................................................... 8

**JOB QUEUES** ................................................................................................................... 8
  - Queues available to all users ........................................................................................ 9
  - Accessing GPU resources ............................................................................................ 9
  - Interactive queue .......................................................................................................... 9
    - Use a GUI interactively ............................................................................................... 10
  - Queue resource limits .................................................................................................. 10
  - Strategic allocation queues ....................................................................................... 11

**COMPILING ON ARTEMIS** .............................................................................................. 12
  - PGI ............................................................................................................................... 12
  - Intel ............................................................................................................................. 12
  - GNU ............................................................................................................................ 13
  - GPU ............................................................................................................................. 13

**RESEARCH DATA STORE (RDS)** ...................................................................................... 14
  - RCOS ........................................................................................................................... 14
    - Transferring data between RCOS and Artemis ....................................................... 14
    - Transferring data from your computer to RCOS ...................................................... 14
  - Classic RDS .................................................................................................................. 16
    - Transferring data between Classic RDS and Artemis ............................................ 16
    - Transferring data from your computer to Classic RDS .......................................... 16

**GLOSSARY** ......................................................................................................................... 17
Getting Started

Access to Artemis

Access to Artemis is available to all University of Sydney researchers and research students who are participating in an approved research project.

Artemis is designed to complement existing HPC facilities at the University. There are also state and national peak facilities available to University of Sydney researchers. These are the National Computational Infrastructure (NCI) and Intersect.

To qualify for access to Artemis, you need to:
- have a valid University of Sydney UniKey,
- be part of an approved research project with an approved Research Data Management Plan (RDMP). You can submit a new RDMP or update an existing RDMP by logging into the RDMP tool.

Once you have submitted your RDMP you will receive notifications indicating that account creation is in progress - one for HPC and a separate one for RDS (the research data store). When your account is active you will receive a welcome email with information to help you get started.

If you have an existing RDS account and need access to Artemis

If you already have an existing RDS account and an approved RDMP, you need to log into the RDMP tool and edit your project on the HPC tab to request access to Artemis:

1. In the Manage area, select Edit
2. Select the HPC tab
3. Check Yes for HPC
4. Resubmit the form

Connecting to Artemis

Artemis supports single sign on from the command line using your UniKey and password. To log in to Artemis from the command line use the hostname hpc.sydney.edu.au

Note: You must be connected to the University network to access Artemis. If you are off campus you will need to use a VPN connection.

For window based access you can use a SSH client like PuTTY.

On Linux or MacOS you can use the SSH command at the command level in the application Terminal. The following is an example of using a SSH command to log into Artemis from a terminal window:

```
ssh <UniKey>@hpc.sydney.edu.au
```

where <UniKey> is your UniKey. If your UniKey is abcd1234, the command to log into Artemis is:

```
ssh abcd1234@hpc.sydney.edu.au
```
You can use the login nodes to gain access to the system, edit files, compile code, analyse data and create job requests for the batch system. They are not intended for computationally intensive processes. Submit all computationally intensive jobs (anything that takes longer than 20 seconds) to the PBS queue.

**Artemis compute nodes**

Artemis is a 4264-core HPC system comprised of the following nodes:

<table>
<thead>
<tr>
<th></th>
<th>Artemis Phase 1 (Existing) Nodes</th>
<th>Artemis Phase 2 (New) Nodes</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Standard Memory</td>
<td>High Memory</td>
</tr>
<tr>
<td>Number of Nodes</td>
<td>56</td>
<td>2</td>
</tr>
<tr>
<td>Cores per node</td>
<td>2 x 12</td>
<td>2 x 12</td>
</tr>
<tr>
<td>RAM per node</td>
<td>128 GB</td>
<td>512 GB</td>
</tr>
<tr>
<td></td>
<td>GPU</td>
<td>GPU</td>
</tr>
<tr>
<td></td>
<td>2 x 12 CPUs 2 GPUs</td>
<td>2 x 16</td>
</tr>
<tr>
<td></td>
<td>128 GB</td>
<td>128 GB</td>
</tr>
<tr>
<td></td>
<td>128 GB</td>
<td>128 GB</td>
</tr>
<tr>
<td></td>
<td>Standard Memory</td>
<td>High Memory</td>
</tr>
<tr>
<td>Number of Nodes</td>
<td>80</td>
<td>3</td>
</tr>
<tr>
<td>Cores per node</td>
<td>2 x 16</td>
<td>4 x 16</td>
</tr>
<tr>
<td>RAM per node</td>
<td>128 GB</td>
<td>6 TB</td>
</tr>
</tbody>
</table>

**Node allocations**

Some nodes of Artemis are restricted to certain groups:

- Civil Engineering have ownership of 13 nodes (416 cores); they have exclusive access to these nodes.
- Some projects were granted strategic allocations on Artemis. These nodes are reserved for these projects. At the launch of the second phase of Artemis, 832 cores are reserved for strategic allocations.

The remaining nodes (3016 cores in total) are available to all users of the system.

**Data Storage on Artemis**

Artemis uses a Lustre file system that is accessible from any compute node and provides excellent I/O performance (especially for large files). The Lustre file system contains the following storage spaces:

- **Home directories** (/home/<UniKey>)
- **Project space** (/project/<Project>)
- **Scratch space** (/scratch/<Project>)

Where <UniKey> is your UniKey and <Project> is your short project name as defined in your RDMP. For example, if your UniKey is abcd1234, then your home directory is /home/abcd1234. If your project name is TEST, then your project space and scratch space directories are /project/TEST and /scratch/TEST, respectively.

Data will not be backed up so we cannot offer any ability to retrieve lost data. For long term storage please store all data in the Research Data Store (RDS).
/home

Each researcher will be allocated their own home directory within Artemis. This can be used to store program code, batch scripts and other files. Please note that the home directory has limited space and is intended for storing code and configuration information. Data on /home is backed up in case of system failure (that is, if the lustre filesystem fails), but files cannot be recovered if they are accidentally deleted. Important data should be stored in the research data store.

To check your /home quota usage, run the following command:

$ lfs quota -h /home/<UniKey>

for example, if your UniKey is abcd1234, you would type:

$ lfs quota -h /home/abcd1234

/project

The default allocation is 1 TB of project storage, shared between all members of the project. Please note that no additional space will be provided. If you require more disk space for job output, use /scratch instead. Since /project (or /scratch or /home) is not backed up, please back up all important data to the Research Data Store regularly.

Your project space is located in the following directory:

/project/<Project>/

where <Project> is your short project name. For example, if your project name is TEST, then your project directory is /project/TEST.

To check your quota and usage of your allocated storage on /project, use the following command:

$ lfs quota -hg <LongProjectName> /project

where <LongProjectName> is your long project name, usually of the form: RDS-<FAC>-<ShortProjectName>-RW, where <FAC> is your three letter faculty code (for example, the code for the Faculty of Science is FSC) and <ShortProjectName> is the name you provided in your RDMP. For example, a project called TEST in the Faculty of Science would have the following long project name: RDS-FSC-TEST-RW.

/scratch

Artemis scratch space is a pool of storage for use during a job. The space is available to everyone on Artemis and there are no restrictions on the amount of data that can be stored. This directory is intended for data that needs to be saved during a job, but can be deleted once the job completes. As this directory is shared by all users, please do not leave your data in this area for longer than absolutely necessary. Transfer important files to /project if you need them after your job finishes. /scratch is NOT backed up and data is periodically removed.

The /scratch directory has approximately 115 Terabytes of space. To see how much is available, use the following command:
$ lfs df -h /scratch

To see how much space your group is using:

$ lfs quota -hg <LongProjectName> /scratch

where <LongProjectName> is your long project name, usually of the form: RDS-<FAC>-<ShortProjectName>-RW, where <FAC> is your three letter faculty code (for example, the code for the Faculty of Science is FSC) and <ShortProjectName> is the name you provided in your RDMP. For example, a project called TEST in the Faculty of Science would have the following long project name: RDS-FSC-TEST-RW.

Software/Programs/Applications

Software required for your work is loaded via module files. These files are a convenient way of rapidly gaining access to software you require.

To see what modules are available for use on Artemis, use the following command:

module avail

this will return a (long) list of modules available for use on the system. To load a module, for example, Matlab, use the following command:

module load matlab

this will add Matlab to your PATH and load any required dependencies. It by default will load the most commonly used version of Matlab. If you require a different version of Matlab, you can specify the version you require as follows:

module load matlab/R2014b

Having multiple versions of the same program open can cause conflicts. For example, if you have used the module load command to load Matlab 2014 and Matlab 2015, then the Matlab command will execute the Matlab executable corresponding to the last module version loaded. For this reason, it is good practice to unload any modules that you do not require:

module unload matlab/R2014b

If you simply want to switch versions of a program, such as versions of Matlab or different versions of the Intel compiler suite, you may use this command instead:

module switch matlab/R2014b matlab/R2015b
Submitting Jobs to Artemis

Artemis uses a modified version of PBS Pro to manage and queue jobs. Some minimal example scripts are provided below.

Serial jobs (1 core)

```bash
#!/bin/bash
#PBS -P MyProject  # your project name
#PBS -l select=1:ncpus=1:mem=4GB  # select one chunk with 1 core and 4 GB memory
#PBS -l walltime=10:00:00  # actual time your job will run for
#PBS -q defaultQ  # place the job in the queue called defaultQ

cd $PBS_O_WORKDIR
<commands to run your computation>
```

You would then submit this to the PBS scheduler using the following command on the command line (assuming you saved your PBS script to a file called MyScript.pbs):

```
$ qsub MyScript.pbs
```

There are other `#PBS` directives that you may optionally use. For a full list, see the PBS Professional user manual.

Parallel jobs using OpenMP

OpenMP is a shared memory parallelism model, so OpenMP parallelised programs can only be run on one chunk (a “chunk” is a virtual node). On Artemis, this means that the maximum number of chunks you can utilise with OpenMP is one and the maximum number of cores is either 24, 32 or 64.

```bash
#!/bin/bash
#PBS -P MyProject  # your project name
#PBS -l select=1:ncpus=4:mem=4GB  # select one chunk with 4 CPUs and 4 GB memory
#PBS -l walltime=10:00:00  # actual time your job will run for
#PBS -q defaultQ  # place the job in the queue called defaultQ

cd $PBS_O_WORKDIR
<commands to run your computation>
```

**Note:** `OMP_NUM_THREADS` is automatically set to `$PBS_NCPUS` on Artemis, so you don’t have to manually set the value of `OMP_NUM_THREADS` in your PBS script.

Parallel jobs using MPI

MPI (Message Passing Interface) is a method of parallelisation that allows jobs to communicate across several nodes, which means your job can use more CPUs than a program parallelised using OpenMP. A basic MPI job script is shown below. This script assumes you are using Intel’s MPI implementation. If your program uses a different MPI implementation, you will need to load module files for that implementation instead of the ones shown here.
#!/bin/bash
#PBS -P MyProject
# your project name
#PBS -l select=1:ncpus=4:mpiprocs=4:mem=4GB
# select one chunk with 4 cores, all of which are available to MPI, and 4 GB memory
#PBS -l walltime=10:00:00
# actual time your job will run for
#PBS -q defaultQ
# place the job in the queue called defaultQ
cd $PBS_O_WORKDIR
# change to current working directory
module load intel-mpi
# add Intel MPI libraries to your $PATH
<commands to run your computation>

The above file will choose one chunk with 4 cores and 4 GB of memory. All 4 cores have been made available to MPI as specified by the mpiprocs=4 option. It is recommended to set ncpus and mpiprocs to the same value, unless you know you can set them differently.

With MPI, you can select more than one chunk:

#!/bin/bash
#PBS -P MyProject
# your project name
#PBS -l select=5:ncpus=8:mpiprocs=8:mem=4GB
# select 5 chunks with 8 cores and 4 GB memory per chunk
#PBS -l walltime=10:00:00
# actual time your job will run for
#PBS -q defaultQ
# place the job in the queue called defaultQ
cd $PBS_O_WORKDIR
# change to current working directory
module load intel-mpi
# add Intel MPI libraries to your $PATH
<commands to run your computation>

This script requests 5 chunks with 8 cores (all 8 of which are available to MPI) and 4 GB of memory per chunk, meaning your job will be allocated 40 MPI cores and 20 GB of memory in total.

Job monitoring/management commands

There are a number of commands available to view the status of jobs on the system. A brief set of useful commands is shown below. For more commands, see the PBS Professional user manual.

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>qstat -u &lt;unikey&gt;</td>
<td>show status of your jobs only</td>
</tr>
<tr>
<td>qdel &lt;jobid&gt;</td>
<td>delete job from queue</td>
</tr>
<tr>
<td>qstat</td>
<td>show status of all jobs</td>
</tr>
<tr>
<td>qstat -f &lt;jobid&gt;</td>
<td>show detailed stats for a specific job</td>
</tr>
<tr>
<td>qstat -xf &lt;jobid&gt;</td>
<td>show detailed stats for a job that finished</td>
</tr>
</tbody>
</table>
Job Scheduling Priority: “Fair Share”

“Fair Share” is a method of assigning priority to jobs based on recent usage of the system. If a project has recently used a lot of CPU time on Artemis, then the priority of their future jobs, relative to other projects, will be reduced. Once a job runs, it is allowed to complete and is unaffected by the fair share system.

Fair Share only has an impact when a capacity limit has been reached and there is contention for resources. Fair Share is calculated at a project level, so if one member of a project uses a lot of CPU time, all jobs submitted by anyone in that project will have lower priority.

Different queues (see the Job Queues section for a description of each queue) will increase your project’s fair share weight faster than others. The small, normal and large queues have a fair share weight of 10, which is considered to be the “standard” fair share weighting. The high memory and GPU queues have a fair share weight of 50. If you request excessive resources (for example, too much memory), your job may be placed in a queue with a higher fair share weighting. Therefore, it is in your best interests to only request resources your job needs.

If you stop using the system, your fair share weight will slowly decrease. The “half-life” of your fair share weight is 2 weeks, which means your fair share weight will decrease by 50% every two weeks (assuming you did not submit any jobs in those two weeks). The exact value of your fair share weight is updated frequently (not just once every two weeks).

Job Queues

The qstat -Q command will show a range of queues on the system. Not all of these are directly available to Artemis users. Some queues are reserved for users with strategic allocations of compute resources. Your job may be routed to other “internal” queues after submission depending on the size of your job.

Queues are requested using the following PBS directive:

```
#PBS -q <QueueName>
```

Where `<QueueName>` is the name of the queue you want to place your job in. For example, if you wanted to place your job into the queue called “defaultQ”, you would write:

```
#PBS -q defaultQ
```
Queues available to all users

The queues available to everyone on Artemis are listed in the following table:

<table>
<thead>
<tr>
<th>Queue</th>
<th>Description</th>
</tr>
</thead>
</table>
| defaultQ      | If you don’t specify a queue, your job will be placed in this queue. Once placed in this queue, your job will be routed to one of the following internal queues based on the resources requested in your PBS script. The sub-queues of this queue are:  
  - Small  
  - Normal  
  - Large  
  - High Memory  
  - GPU  
  For details of resource limits of these queues, see the defaultQ resource limits. |
| small-express | An "express" queue for small jobs. These jobs will typically start quickly, but will increase your fair share weight 5 times faster than jobs in defaultQ. This queue is useful for quickly testing your jobs before submitting a longer job to defaultQ. |
| scavenger     | A special, low priority queue that attempts to run jobs on idle resources if and when they are available. If your job completes it will be “free”, in that it will not increase your fair share weight at all. The downside is there is no assurance that your job will start. If the system is heavily utilised it may never run. Additionally, if a job in any other queue requires resources that a scavenger job is using, the scavenger job will be suspended. If the scavenger job cannot be resumed within a given period (approximately half the expected run time) it will be terminated and you will lose all unsaved progress. |

Accessing GPU resources

To use GPUs (Graphical Processing Units), you simply specify your GPU requirements as part of select statement. For example:

```
#PBS -l select=1:ncpus=1:ngpus=1
ngpus should be 1 or 2 per chunk as GPU nodes have a maximum of 2 GPUs.

Note: The fair share weighting of this queue is 5 times higher than defaultQ.
```

Interactive Queue

Artemis has two nodes dedicated to running jobs interactively. You can request an interactive jobs using the qsub -I directive from the command line. For example:

```
$ qsub -I -l -P <ShortProjectName> select=1:ncpus=1:mem=4GB,walltime=1:00:00
```

where <ShortProjectName> is the short project name you specified in your RDMP. If your project name is TEST, then the above qsub command is:

```
$ qsub -I -l -P TEST select=1:ncpus=1:mem=4GB,walltime=1:00:00
```
Note: Interactive jobs can only be requested on the command line, as shown above. You cannot use a job script to gain access to compute nodes interactively.

**Use a GUI interactively**

One use of the interactive queue is to use a program with a Graphical User Interface (GUI) interactively. To use a program with a GUI on a compute node on Artemis requires that you are running an X Server on your computer. On Windows, we recommend Xming and on MacOS we recommend XQuartz. Once your X Server is running, log into Artemis using the SSH login command with X Forwarding:

```
$ ssh -X <UniKey>@hpc.sydney.edu.au
```

Then, request an interactive session using `qsub -I -X`. An example `qsub` command is:

```
$ qsub -I -X -l -P <ShortProjectName> select=1:ncpus=1:mem=4GB,walltime=1:00:00
```

Users with strategic allocations can run interactive jobs in their allocated area by specifying a queue in the interactive `qsub` command. For example, members of the project “RDS-FEI-porous-RW” can request an interactive session in their allocated area with the following command:

```
$ qsub -I -X -l -P RDS-FEI-porous-RW -q condo -civil select=1:ncpus=1:mem=4GB,walltime=1:00:00
```

**Queue resource limits**

<table>
<thead>
<tr>
<th>Queue</th>
<th>Max Walltime</th>
<th>Max Cores per User</th>
<th>Memory per node</th>
<th>Fair Share Weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>Small</td>
<td>1 day</td>
<td>96</td>
<td>&lt; 125 GB</td>
<td>10</td>
</tr>
<tr>
<td>Normal</td>
<td>7 days</td>
<td>96</td>
<td>&lt; 125 GB</td>
<td>10</td>
</tr>
<tr>
<td>Large</td>
<td>21 days</td>
<td>600</td>
<td>&lt; 125 GB</td>
<td>10</td>
</tr>
<tr>
<td>High Memory</td>
<td>7 days</td>
<td>192</td>
<td>125 GB to 6 TB</td>
<td>50</td>
</tr>
<tr>
<td>GPU</td>
<td>7 days</td>
<td>24</td>
<td>&lt; 125 GB</td>
<td>50</td>
</tr>
<tr>
<td>small-express</td>
<td>12 hours</td>
<td>96</td>
<td>&lt; 125 GB</td>
<td>50</td>
</tr>
<tr>
<td>scavenger</td>
<td>2 days</td>
<td>288</td>
<td>&lt; 125 GB</td>
<td>0</td>
</tr>
<tr>
<td>Interactive</td>
<td>4 hours</td>
<td>4</td>
<td>&lt; 125 GB</td>
<td>100</td>
</tr>
</tbody>
</table>

- The Small, Normal, Large, High Memory and GPU queues are all accessed via the defaultQ queue. You cannot directly request these queues.
- The **interactive queue** is requested using the `qsub -I` command via the command line. You cannot request interactive access with `#PBS -q interactive`.
- The maximum number of jobs a user can have queued is:
  - 200 in defaultQ,
  - 10 in small-express.
- PBS array jobs are limited to 1000 elements.
Strategic Allocation Queues

For users with strategic allocations can run their jobs in their allocated areas by requesting the appropriate queue:

<table>
<thead>
<tr>
<th>Queue</th>
<th>Description</th>
</tr>
</thead>
</table>
| condo-civil | This queue grants access a set of nodes reserved for Civil Engineering. Only projects authorised by Civil Engineering can specify this queue. At launch they are:  
- RDS-FEI-porous-RW  
- RDS-FEI-pyphd-RW  
- RDS-FEI-R4SPH-RW  
If you are in the School of Civil Engineering, please speak to Luming Shen for access.                                                                 |
| alloc-dh    | This queue requests access to resources reserved for users in the project called “MetroScan-TI”, which is led by Prof. David Hensher. If a user is not a member of the project, and does not specify the right project for the queue, their job will be rejected. |
| alloc-jr    | This queue requests access to resources reserved for users in the project called “Gene and Stem Cell Therapy Program”, which is led by Prof. John Rasko. If a user is not a member of the project, and does not specify the right project for the queue, their job will be rejected. |
| alloc-nw    | This queue requests access to resources reserved for users in the project called “Investigation of the critical factors affecting mixing and water quality in density stratified riverine flows”, which is led by Dr. Nicholas Williamson. If a user is not a member of the project, and does not specify the right project for the queue, their job will be rejected. |
| alloc-am    | This is an allocation assigned to the project called “Computational Multi-Scale Approach for Process Intensification and Innovation” which is led by Dr. Alejandro Montoya. If a user is not a member of the project, and does not specify the right project for the queue, their job will be rejected. |
| alloc-md    | This is an allocation assigned to the project called “Rational Materials Design: Gas Adsorption in Porous Crystals” which is led by A/Prof. Meredith Jordan. If a user is not a member of the project, and does not specify the right project for the queue, their job will be rejected. |

**Note:** Users with strategic allocations are also free to request their jobs be run in defaultQ, small-express and scavenger queues in addition to their strategic allocation queue.
## Compiling on Artemis

Researchers using the Artemis service have access to the following compiler suites to compile and debug their own code:

- Portland Group (PGI)
- Intel
- GNU
- GPU

### PGI

To use the PGI compiler, load the PGI modules with the following command:

```bash
module load pgi
```

or

```bash
module load openmpi-pgi
```

for OpenMPI programs.

The following table shows the various commands to use when compiling code, depending on the language and style of execution being used:

<table>
<thead>
<tr>
<th>Language</th>
<th>Single CPU</th>
<th>Using MPI</th>
<th>Using OpenMP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fortran</td>
<td>pgf90</td>
<td>mpif90</td>
<td>pgf90 -mp</td>
</tr>
<tr>
<td>C</td>
<td>pgcc</td>
<td>mpicc</td>
<td>pgcc -mp</td>
</tr>
<tr>
<td>C++</td>
<td>pgcc</td>
<td>mpicxx</td>
<td>pgCC -mp</td>
</tr>
</tbody>
</table>

### Intel

To use the Intel compilers, load the Intel modules with the following command:

```bash
module load intel
```

or

```bash
module load intel-mpi
```

for Intel MPI programs.

The following table shows the various commands to use when compiling code, depending on the language and style of execution being used:

<table>
<thead>
<tr>
<th>Language</th>
<th>Single CPU</th>
<th>Using MPI</th>
<th>Using OpenMP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fortran</td>
<td>ifort</td>
<td>mpiifort</td>
<td>ifort -openmp</td>
</tr>
<tr>
<td>C</td>
<td>icc</td>
<td>mpicc</td>
<td>icc -openmp</td>
</tr>
<tr>
<td>C++</td>
<td>icpc</td>
<td>mpicxx</td>
<td>icpc -openmp</td>
</tr>
</tbody>
</table>

Intel compiled implementations of OpenMPI are available in the openmpi-intel modules:
module load intel openmpi-intel

**GNU**

To use the GNU compilers, load the GNU compiler modules:

```bash
module load gcc
or
module load gcc openmpi-gcc
```

for OpenMPI programs.

The following table shows the various commands to use when compiling code, depending on the language and style of execution being used:

<table>
<thead>
<tr>
<th>Language</th>
<th>Single CPU</th>
<th>Using MPI</th>
<th>Using OpenMP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fortran</td>
<td>gfortran</td>
<td>mpif90</td>
<td>gfortran -fopenmp</td>
</tr>
<tr>
<td>C</td>
<td>gcc</td>
<td>mpicc</td>
<td>gcc -fopenmp</td>
</tr>
<tr>
<td>C++</td>
<td>g++</td>
<td>mpicxx</td>
<td>g++ -fopenmp</td>
</tr>
</tbody>
</table>

**GPU**

To take advantage of the GPU accelerated compute nodes, your code will need to be modified to run in the hybrid GPGPU environment.

The compile chain will need to specify that the target is GPU via the target argument to the pgcc/pgfortran compiler. This include the compiler argument: `-ta=nvidia`. This compiler flag tells the PGI compiler to compile for the Nvidia GPU.

The preferred method of accessing GPUs is to use the NVIDIA CUDA toolkit:

```bash
module load cuda
```

which places the nvcc compiler in your path. See the [NVIDIA CUDA programming guide](https://nvidia.com) for more details.
Research data store (RDS)

The University of Sydney provides a research data storage service (called RDS) to researchers at the University of Sydney to meet modern data handling requirements (your data must be protected from loss and unauthorised access). Data in the RDS are regularly backed up, have built in redundancy, and are covered by the University’s Information Security Policy.

There are two different RDS services: Classic RDS and Research Computing Optimised Storage (RCOS). Of these two, RCOS is the easiest to use with Artemis.

RCOS

Transferring data between RCOS and Artemis
RCOS is a linux-based file server, and is therefore directly accessible from Artemis login nodes in the directory:

```
/rds/PRJ-<ShortProjectName>
```

where `<ShortProjectName>` is your project’s short project name, as defined in your RDMP. For example, if your short project name is TEST, then your RCOS storage is in the directory:

```
/rds/PRJ-TEST
```

If you are logged into Artemis, you can use standard linux shell commands, such as `cp` and `mv` to move data to and from Artemis and RCOS.

Note: Only the Artemis login nodes can access `/rds`. If you need to analyse data on Artemis, you must transfer your data from `/rds` to either `/project` or `/scratch` before submitting your job.

Transferring data from your computer to RCOS
If you wish to transfer data from your own personal computer to RCOS, you need to use SFTP (Secure File Transfer Protocol). RCOS is accessible via SFTP with the following command:

```
sftp <UniKey>@rcos.sydney.edu.au
```

Where `<UniKey>` is your UniKey. For example, if your UniKey is abcd1234, then the command would be:

```
sftp abcd1234@rcos.sydney.edu.au
```

If you are on-campus, you can use this address to access RCOS:

```
sftp <UniKey>@rcos-int.sydney.edu.au
```

The next step is crucial:

```
cd /rds/PRJ-<ShortProjectName>
```

this directory is the only directory that Artemis can access. Save all your data here. Don’t save your data in `/home`. If your data is saved in `/home`, move it to `/rds/PRJ-<ShortProjectName>`.
If you prefer to transfer data using a program with a graphical user interface, you can use programs such as CyberDuck or FileZilla.

**Note:** the directory `/rds/PRJ-<ShortProjectName>` is not visible in the FileZilla or Cyberduck directory tree.

Instead, you must manually type `/rds/PRJ-<ShortProjectName>` into the remote directory box to access your RCOS storage area.
Classic RDS

Transferring data between Classic RDS and Artemis

Classic RDS is a Microsoft Windows file server, therefore you will need to use smbclient to move data from Artemis to Classic RDS:

```
smbclient //research-data.shared.sydney.edu.au/<VolumeName> -U UniKey -W SHARED
```

Where `<VolumeName>` is the name of the volume where your data is located and `<UniKey>` is your UniKey. If your UniKey is abcd1234 and your data is stored in the volume FSC, the command would be:

```
smbclient //research-data.shared.sydney.edu.au/FSC -U abcd1234 -W SHARED
```

If you successfully logged in, your command prompt will change to this:

```
smb: />
```

You can move around the filesystem in smbclient in the same way as you do on Artemis. To push files from Artemis to Classic RDS, use the `put` command. To pull files from Classic RDS to Artemis, use the `pull` command. For example, to put a file called `my_important_data.out` onto Classic RDS, use the command (after logging into smbclient):

```
smb: /> put my_important_data.out
```

To run local shell commands (that is, to navigate around Artemis's file system) while logged into smbclient, place an exclamation mark before your shell command. For example, to view files in your current directory on Artemis, type:

```
smb: /> !ls
```

Transferring data from your computer to Classic RDS

If you are on Windows or MacOS and on-campus, you can access Classic RDS by following the instructions here. (http://staff.ask.sydney.edu.au/app/answers/detail/a_id/174/~/how-do-i-map-my-network-drive-so-i-can-access-my-network-storage%3F) You can then browse and transfer your files using the native MacOS or Windows file explorers.
<table>
<thead>
<tr>
<th>Name</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chunk</td>
<td>A virtual computer. It is possible to have multiple virtual computers on one physical computer.</td>
</tr>
<tr>
<td>CI</td>
<td>Chief Investigator</td>
</tr>
<tr>
<td>Classic RDS</td>
<td>Windows fileserver that is backed up regularly</td>
</tr>
<tr>
<td>Core/CPU</td>
<td>Central Processing Unit. Technically these two are different, but they are used interchangeably in this document.</td>
</tr>
<tr>
<td>GNU</td>
<td>An extensive collection of open source software, including compilers.</td>
</tr>
<tr>
<td>GPU</td>
<td>Graphics Processing Unit</td>
</tr>
<tr>
<td>HPC</td>
<td>High Performance Computing</td>
</tr>
<tr>
<td>Intel</td>
<td>Intel Compiler Suite</td>
</tr>
<tr>
<td>Intersect</td>
<td>Intersect High Performance Computer</td>
</tr>
<tr>
<td>Lustre</td>
<td>A high performance filesystem</td>
</tr>
<tr>
<td>MPI</td>
<td>Message Passing Interface</td>
</tr>
<tr>
<td>NCI</td>
<td>National Computational Infrastructure</td>
</tr>
<tr>
<td>Node</td>
<td>A physical computer. Artemis is comprised of many nodes.</td>
</tr>
<tr>
<td>OpenMP</td>
<td>A shared memory parallelism application programming interface</td>
</tr>
<tr>
<td>OpenMPI</td>
<td>An open source implementation of Message Passing Interface</td>
</tr>
<tr>
<td>PBS</td>
<td>Portable Batch System</td>
</tr>
<tr>
<td>PGI</td>
<td>Portland Group Compiler Suite</td>
</tr>
<tr>
<td>PuTTY</td>
<td>A secure shell client for Windows</td>
</tr>
<tr>
<td>RAM</td>
<td>Random Access Memory</td>
</tr>
<tr>
<td>RCOS</td>
<td>Research Computing Optimised Storage - A Linux fileserver that is backed up regularly</td>
</tr>
<tr>
<td>RDMP</td>
<td>Research Data Management Plan</td>
</tr>
<tr>
<td>RDS</td>
<td>Research Data Store</td>
</tr>
<tr>
<td>SFTP</td>
<td>Secure File Transfer Protocol</td>
</tr>
<tr>
<td>SSH</td>
<td>Secure Shell</td>
</tr>
<tr>
<td>Terminal</td>
<td>A command line interface to a computer</td>
</tr>
<tr>
<td>UniKey</td>
<td>Your Sydney University Unique ID</td>
</tr>
<tr>
<td>VPN</td>
<td>Virtual Private Network</td>
</tr>
</tbody>
</table>