Access to HPC software

All HPC facilities, worldwide, provide software in a very different way to personal and corporate computing platforms. JCU HPC uses environment modules to deliver multiple versions of software to researchers using our multi-user HPC cluster. This approach is widely used for the following reasons:

- Most operating systems weren't written to deal with having multiple versions of the same software available. Even in cases where multiple versions are installed, e.g., java, it is left to a system administrator to set the version which everyone will use.
- Using environment modules, Individual users or groups can determine which version of software they wish to use.
- Loading of modules with version information maximizes your chance of being able reproduce your results upon demand.
- In the corporate world, system administrators have to configure a default version of piece of software (e.g., java) that everyone will use.
- Using environment modules also improves performance by reducing the length of your search path (executable of interest found sooner). It also reduces the chance of you having to deal with application name conflicts.

In 2021, HPC staff commenced a major project to replace all existing software (including conda environments) with containerized versions of the same software and/or environments. The container delivery platform being using is Apptainer (singularity) which mirrors the approach that all other HPC facilities are using or moving toward using. The move to containerized HPC is being driven by a requirement for all research to be done in a reproducible way.

For security reasons, all software installed as part of the OS must be kept up to date.

- This has potential to impact your computational research results.
- Most researchers would be unaware of the behind the scenes changes or what impacts they might have had.
- It would be nigh on impossible for me to recreate the state of HPC cluster operating systems at any nominated point in time.

Software installed using the traditional, compile from source, method comes with serious consequences to reproducible research.

- Such installs usually result in a software environment with a potentially very large number of unknown dependencies.
- Once again, this makes it nigh on impossible to reproduce at a later date.

Operating system and software providers release software with a defined lifecycle.

- When RedHat Enterprise Linux 6 (RHEL6) went end of life, we moved to RHEL7. All software had to be recompiled/reinstalled.
- Obtaining a copy of RHEL6 would be very difficult. Being given security clearance to use it would be even more difficult.
- After python 2 hit end of life a few environments that required python 2, indirectly, could not be reinstalled (dependency not available).

In many cases, the above points mostly relate to risk of work not being reproducible. Moving to containerized delivery of software provides a guarantee of reproducibility, assuming that the container platform continues to be available and supported.

Environment Modules Cheat Sheet.

Information Requests

You will need to replace any occurrence of `<software-name>` and/or <version> below with an appropriate name/value.

Listing all available software

```
module avail
```

Listing all versions of a given piece of software

```
module avail <software-name>
# A partial match will be sufficient.
```

Display a brief summary about a given piece of software

```
module help <software-name>
```

Find out how your environment will change after loading a module

```
module show <software-name>/<version>
# For apptainer based software, you will see aliases that have been created for ease of use.
```

List environment modules that have been loaded already

```
module list
```

Modification of your environment (paths & aliases)

You will need to replace any occurrence of `<software-name>` and/or <version> below with an appropriate name/value.

To enable your access to a specific software environment

```
module load <software-name>/<version>
# Omission of "/<version>" will result in the default version (changes over time) being loaded.
# Note that the default version may be the "safest", rather than the latest, version.
```

To switch to use of another version of the same piece of software

```
module switch <software-name>/<version>
```

To unload a module you previously loaded

```
module unload <software-name>
# The specification of a version shouldn't be required.
```

To unload all modules you previously loaded

```
module purge
```

Information for researchers using conda environments

Environment module files for conda environments have been removed, since discovery of a few conda environments that couldn't be reinstalled. After you issue the command

```
module load conda3
```

you will be able to use the conda command to list environments available.

Notes for future

Conda environments will not be available on future HPC platforms.

There is increasing pressure to move HPC workloads into the public cloud. Reliance on in-house HPC infrastructure will decrease with time.

It's possible that HPC will be directed to implement a NOEXEC option on all user filesystems, which would mean software you have installed yourself will not be executable. I would hope that there is a documented security vetting process prior to such a directive being given to us. We have had a request for implementation of such an option in the past but chose not to take any action (there was no directive).