GWDG
AG Computing
Artur Wachtel

Exercise 1 / October 20, 2023
HPC System Administration / WiSe 2023/24
40 Minutes Total

Exercise Introduction

In this exercise we will install Lmod and Spack on your cluster. Then we use these tools to install additional software and provide environment modules, so users can easily load the software they need.

Task 1: Preparations (5 min)

Make sure you have the EPEL repository active and install the dependencies for Lmod.

```
$ sudo yum install -y epel-release
$ sudo yum install lua lua-devel lua-json lua-lpeg lua-posix lua-term lua-filesystem
$ sudo yum install tcl tcl-devel
```

The above packages (except the `-devel` versions) are required **on the node image too**.

Afterwards, you can already install the dependencies for spack.

```
$ sudo yum install curl findutils gcc-c++ gcc gcc-gfortran git gnupg2 hostname
$ sudo yum install iproute redhat-lsb-core make patch
$ sudo yum install python3 python3-pip python3-setuptools unzip
```

We will use a separate user account with limited access rights for the management of software.

```
$ sudo useradd -m swmanager
$ sudo mkdir /opt/sw
$ sudo chown swmanager:swmanager /opt/sw
```

Open a login shell for the new user account.

```
$ sudo su - swmanager
```

For the remainder of this exercise sheet, we always use the new user `swmanager` unless the command starts with `sudo`.

Hints

- If you want to test this setup on your cluster, make sure `/opt` is a shared NFS mount. Furthermore, make sure that the user account `swmanager` is synced into the node images.
  ```
  $ sudo wwctl container syncuser --write rocky-8
  $ sudo wwctl overlay build
  ```

Task 2: Install Lmod (5 min)

Download the source for Lmod via e.g. `wget https://github.com/TACC/Lmod/archive/refs/tags/8.7.19.tar.gz`
Extract the tarball and install Lmod into `/opt/sw/` via:

```
$ tar -xf 8.7.19.tar.gz
$ cd Lmod-8.7.19/
$ ./configure --prefix=/opt/sw
$ make install
```

Finally, to make Lmod available for all users, we have to create a symbolic link into the profile folder.

```
$ sudo ln -s /opt/sw/lmod/lmod/init/profile /etc/profile.d/modules.sh
```

This last step has to happen **on the node image, too**.

Afterward, you can log out and log in to get a shell environment with the new profile script. From now on, every user has acces to the module system.

```
$ module avail
```

Task 3: Install Spack (5 min)

Assuming you have already installed the dependencies for spack as per Task 1, we can just download spack and extract the tarball.
`https://github.com/spack/spack/releases/download/v0.19.1/spack-0.19.1.tar.gz`
No compilation is required.

Afterward, copy a default configuration file, so we can modify it freely.

```
$ cd spack-0.19.1/etc/spack
$ cp defaults/config.yaml .
```

Modify the new `config.yaml` such that `install_tree:root` is `/opt/sw/spack`.

Next, create a new file with name `modules.yaml` in this very folder, its content shall be

```
modules:
  default:
    enable:
      - lmod
    roots:
      lmod: /opt/sw/modules
    lmod:
      hash_length: 0
      hierarchy:
        - mpi
      core_compilers:
        - gcc@8.5.0
```

Return to your home and activate Spack by sourcing the setup script.

```
$ cd ~
$ source spack-0.19.1/share/spack/setup-env.sh
```

Finally, we need to tell Lmod where to find the packages from spack:

```
$ echo '/opt/sw/modules/linux-centos8-x86_64/Core' > /opt/sw/lmod/lmod/init/.modulespath
```

Task 4: Install a first package (5 min)

Let us start with something that we can re-use later: `tar`. We first look at the detailed info of that package, inspect the concretized specification with its dependency tree, and finally install it.

```
$ spack info tar
$ spack spec tar
$ spack install tar
```

Task 5: Install Gromacs (5 min)

This task is optional and will **spend a long time compiling** the actual software and its dependencies.

Inspect its dependency tree first. All the dependencies that we have already installed start with `[+]`.

```
$ spack spec -I gromacs
$ spack install gromacs
```

While this installation process continues, you can open a new terminal window, log into your cluster and continue with the remaining tasks. Keep an eye on the installation process in case of error messages.

Task 6: The module command (5 min)

After Task 4, our system has two different verisons of `tar`. Let us see if we can dynamically switch between the two. If the configuration of Lmod and Spack is correct, `spack install` has created module files that we can load with Lmod. Make sure you understand what each command does:

```
$ tar --version
$ which tar
$ module avail tar
$ module load tar
$ tar --version
$ which tar
$ module unload tar
$ tar --version
$ which tar


$ module purge
```

Hints

- When in doubt, you can always check `$ module --help`.

Task 7: Spack specifications (10 min)

How are the following specifications to be interpreted? When in doubt, consult `spack info`.

```
tar zip=gzip ^bzip2+debug ^xz+pic
gromacs+blas+cuda
gromacs%gcc@12.2.0 ^openmpi%gcc@8.5.0
```

Use `spack spec -I` to check if you could install the above specs, and which previously installed packages could be reused.

Use `spack list` to search for a piece of software that you care about, find the package description via `spack info` and look through its options. Formulate a package specification for this piece of software and validate the spec via `spack spec`. Are you surprised how many dependencies the package has?

Optional Task 8: Module Hierarchies (10 min)

This is a difficult **additional** task which will support your understanding in the topic.

Assuming Task 5 was successful and you have installed gromacs via spack, try to locate the module:

```
$ module avail gromacs
$ module spider gromacs
```

The module hierarchy prevents you from loading gromacs directly. Instead you need to first load openmpi:

```
$ module load openmpi
$ module load gromacs
```

It is good practice to provide modules in hierarchies. Thus, modules that are built with conflicting implementations of MPI can coexist on the same system. We can make this more transparent by adding another path to the `.modulespath` for Lmod:

```
$ echo /opt/sw/modules/linux-centos8-x86_64/openmpi/4.1.4-fbbqxfu/Core/ >>
↪  /opt/sw/lmod/lmod/init/.modulespath
```

It is possible that your Spack compiled a slightly different configuration, which means the hash after the version of openmpi might be different in your case.

Optional Task 9: Cleaning up the module environment (10 min)

This is a difficult **additional** task which will support your understanding in the topic.

Assuming Task 5 was successful and you have installed gromacs via spack, look at all available modules:

```
$ module avail
```

This list is very cluttered with software that most users will never want to load explicitly. We can modify the configuration of Spack to generate only specific modules.
Let us modify the file `spack-0.19.1/etc/spack/modules.yaml` and extend it by two sections:

```
modules:
  default:
    enable:
      - lmod
    roots:
      lmod: /opt/sw/modules
    lmod:
      hash_length: 0
      hierarchy:
        - mpi
      core_compilers:
        - gcc@8.5.0
      include:
        - openmpi
        - gromacs
      exclude:
        - '%gcc@8.5.0'
```

Thus we explicitly tell Spack to provide module descriptions for openmpi and gromacs, yet all other modules that have been compiled via `gcc@8.5.0` will be excluded from the module generation. Finally, we re-create a clean set of module files via

```
$ spack module lmod refresh --delete-tree
$ module avail
```