Running Singularity containers on NCAR systems

Cheyenne and Casper users can run their own Singularity containers on those systems as described on this page. This documentation details which modules to load to support the container, which commands to run, and how to perform a simple test in the new Singularity container environment.

After completing a successful test, users should be able to use their Singularity container on the host system. Performance has not been tested.

Before you start, do the following to prepare:

- Get access to an image in Singularity Image Format (SIF) for your container.
- Create a *Hello, World* code to compile as a test.
- Log in to Cheyenne or Casper.

Step-by-step instructions

Load the CISL Singularity module as shown.

```
module load singularity
```

Create a sandbox. (Use your own container name and .sif file.) The following example generates a sandbox under a directory named **container_name**.

```
singularity build --sandbox container_name container_name.sif
```

If you need to delete a sandbox directory, you will first need to set the proper permissions by executing the **find** command, then run the **rm** command.

```
find container_name -type d -exec chmod 777 {} \;
rm -rf container_name
```

Load a shell in the container.

```
singularity shell -u -B/glade:/glade container_name
```

The default shell is bash. To invoke a different shell – tcsh, for example – specify it as follows:

```
singularity shell --shell /usr/bin/tcsh -u -B/glade:/glade container_name
```

In the container shell, compile your *Hello, World* code to run as a test.

```
mpif90 hello_world.f90
```

Run the executable *in the container*. The following example command will run four tasks on the same node.

```
mpiexec -np 4 ./a.out
```

Exit the container after the job runs.

```
exit
```

For the next step, you will launch and run an MPI command on the host system *outside of the container*, but before doing so, be sure to use the same MPI version on the host and container. Using mismatched MPI libraries or library versions will likely result in a fatal error reported as:

```
asallocash failed: array services not available
mpiexec: all_launch.c:737: newash: Assertion `new_ash != old_ash' failed.
```

In contrast, a successful launch and run will produce the same output as when the executable ran in the container. Here is the MPI command for the next step:

```
mpiexec -np 4 singularity run -u -B/glade:/glade container_name ./a.out
```

When you have confirmed that the output is the same as before, submit an interactive job to run on multiple nodes as in this example, using four nodes with two tasks on each node.

```
qsub -I -l select=4:ncpus=2:mpiprocs=2 -q queue_name -l walltime=02:00:00 -A project_name
```

When the interactive job starts, confirm the Singularity executable is still available by using the **which** command.

```
which singularity
```

Then execute the following command at the new prompt to run the same job on multiple nodes *within the container*. The output from this job (example shown) should be the same as for the previous job.

```
mpiexec -np 8 singularity run -u -B/glade:/glade container_name ./a.out

Hello from 0 / 8 running on r3i6n15
...
Hello from 6 / 8 running on r3i6n14
Hello from 7 / 8 running on r3i6n8
```