What is happening

IT will be updating experimental nuclear science linux systems to Debian 10 (Buster). In order to allow a smooth transition, the experimental software group will be deploying singularity containers that will allow you to continue to run your Debian 8 (jessie) for some time to come, although we do encourage you to migrate to buster.

At present there is a fishtank-like system called 'flagtail' on which we've deployed both buster and jessie containers. In the near future IT will produce one or more buster spdaq like systems as well for DAQ testing.

What action do I need to take

Try out your software within the flagtail containers. Let us (fox@nscl.msu.edu and/or cerizza@nscl.msu.edu) know if software is missing from those containers.

The remainder of this section describes how to run jessie and buster containers mapping any directories you need into the container. We'll also describe the differences you'll need to take into account when taking data on the test spdaq system.

Quick start:

Use the scripts /usr/opt/jessie.sh and /usr/opt/buster.sh as starting points for scripts to start the cointainers for Debian 8 (Jessie) and Debian 10 (Buster) respectively.

The singularity *command:*

The singularity command can be used to run either the jessie or buster containers at /usr/opt/{nscl-buster.img,nscl-jessie.img}. The sample scripts /usr/opt/jessie.sh and /usr/opt/buster.sh are good starting points. These scripts map the correct version of /usr/opt to the container's /usr/opt directory. I encourage you to start by copying this script and then just adding to the bind points.

The singularity command you'll need is:

singularity shell --bind *bindpoint-list image-name*

image-name is the name of one of the singularity images above. *bindpoint-list* is a list of directories you want to make visible inside of your container (note that your home directory tree and current working directory tree are automatically bound. *bindpoint-list* is a comma separated list of elements of the form

native-dir[:*container-dir*]

native-dir is the name of a directory in the 'host' filesystem and *container-dir* is where that directory should appear in the container filesystem. If omitted, *native-dir* appears in the same location in the container as in the host filesystem. You can use this to make /projects subdirectories (e.g.) visible in the container. You can provide as many bindings as you need.

Data acquisition systems:

Data taking requires that processes be run over ssh pipes in local or remote systems. As we move to a fully containerized system, the system nees to know how to rebuild the container and bind points you are using on the other end of its ssh pipes. NSCLDAQ's ssh pipeline software has been modified to detect that it's running in a container and, with some assistance not provided by singularity, reconstruct the local container and bind points on the other end of the SSH pipe.

The script you build to start the container you're using for data taking must:

1. Define the environment variable **SING_IMAGE** to be the full singularity image path (Singularity provides environment variables that name the image but not its full path).
2. Store the bindings you're using, one per line in ~/.singularity_bindpoints.

Here's an example olf a script that does that:

```
export SING_IMAGE=/usr/opt/nscl-buster.img

echo /usr/opt/opt-buster:/usr/opt > ~/.singularity_bindpoints
echo /scratch                    >> ~/.singularity_bindpoints


singularity shell --bind /usr/opt/opt-buster:/usr/opt,/scratch \
    $SING_IMAGE
```

The ssh pipe software will start your e.g. Readout program in the container described by SING_IMAGE (which gets propagated into the container) and with the proper /usr/opt maping as well as /scratch mapped to /scratch. By default, the home directory tree is bound into the container as well.