Container Deployment Baseline

MRI Center, NYSPI

MRI Computing Team

v1 1/2023

The purpose of this document is to establish the baseline features required for Docker containers to be deployed within pipelines on MRImaker and Docker Swarm. Meeting this baseline is the responsibility of the image developer(s) who has technical knowledge of the science functions being performed by the container. Once this threshold has been reached, the MRI Computing Team will work with the developer(s) to integrate this image into a functioning pipeline on MRImaker. Specific standards for image building and internal container security are currently outside the scope of this document, however, all images should meet the security requirements established elsewhere in organizational policy for NYSPI, CUMC, and OMH.

* It should be clear whether the container can run using the local Docker host on MRImaker or requires sufficient resources and runtime to need a Docker Swarm service. This largely depends on how much demand the compute operations being performed by the container will place on the Docker host; please reach out to the MRI Computing Team if you are unsure of how to proceed here. The total expected resource use of all containers should be tested and understood before deployment to production. The MRI Computing Team and psyIT have the final input on resource utilization and will approve new containers based on use case.
* The design of the image should account for the need to remove the resulting container or service as soon as possible after the production run has completed in the final pipeline. Local containers should run with the automatic remove option whenever possible or another mechanism to automatically remove the stopped container. Swarm services should either be automatically removed in the final pipeline or managed with an interface the MRI Computing Team can help develop.
* All process outputs, including intermediate data, logs, and temp files, should be written directly to permanent storage locations mounted into the container whenever possible.
* It is not possible to write valid symbolic links from a container to a storage location mounted within a container. For this reason, it is strongly discouraged to create any containerized processes that depend on symbolic links. However, due to the large number of legacy software used in MRI processing that depends on links, this is allowed when necessary. If a containerized process must use symbolic links, it is required to write all outputs into container space and then use a copy command with archive options enabled to copy them to the mounted storage location before stopping the container.
* Dynamic inputs to the container should be limited to pointers to specific data locations or sessions and operation options that can be flagged using simple bash. Complex options, pipeline configuration, and other variable inputs that cannot be included at runtime in the initial command should be mounted into the container as a configuration file and guidance provided on how to manage these files in a data structure in permanent storage.