Container Support in Pegasus 4.8.x

Karan Vahi
vahi@isi.edu

USC Viterbi
School of Engineering
Information Sciences Institute

https://pegasus.isi.edu

Containers

- Lightweight and a reproducible way to run application on heterogeneous nodes.

- Separates the application from the node OS.

- Popular Container Technologies
  - Docker
    - Popular in the enterprise world.
    - By default, application launched in container run as root
      - A concern when running on shared infrastructure
  - Singularity
    - Popular in HPC environments.
    - Is run in user space.

Why use Containers for your workflow?

Traditional way of referring user executable in Pegasus

- Jobs in the input abstract workflow (DAX) refer to logical transformations.
- Users define mapping of logical transformation to actual executable in a Transformation Catalog

executables description

list of executables locations per site

physical executables

mapped from logical transformations

transformation type

whether it is installed or available to stage

```
...
# This is the transformation catalog. It lists information about each of the
# executables that are used by the workflow.

tr ls {
  site compute-site{
    pfn "/bin/ls”
    type "INSTALLED”
    arch "x86_64"
    os "linux"
    osrelease ”centos"
    osversion “7"
  }
}
...
```

Executable staging works if executable is statically linked , OR

if libraries are installed on the nodes for dynamically linked executables

Why use Containers for your workflow?

Traditional way of referring user executable in Pegasus

- Pegasus matches the attributes of an executable defined in Transformation Catalog against the attributes specified for site in Site Catalog.

- This approach works fine if your site is made of homogenous nodes

- However, problems occur when

  ① you run on a site with hetrogeneous nodes and your job lands on a node where OS is incompatible with your executable

  ② Application is a mis-match to the compute node environment

    - Install libraries in your shared space and make sure environment refers to those libraries
    - Need cooperation from Site Admins. On OSG , you can install things in CVMFS
    - TensorFlow requires specific python libraries and versions. Some libraries maybe easy to install on latest Ubuntu, but not on EL7

Pegasus Container Support

- Introduced in Pegasus Release 4.8
  - Support for both Docker and Singularity

- Users can now refer to containers in the Transformation Catalog

  with their executables preinstalled.

- Users can refer to a container they want to use. However, they can let Pegasus stage their executable to the node.
  - Useful if you want to use a site recommended/standard container image.
  - Users are using generic image with executable staging.

Specifying Containers in Transformation Catalog

```
...
tr pegasus::keg{

  site isi {
    pfn "/usr/bin/pegasus-keg
    arch "x86"
    os "linux"
    osrelease ”centos"
    osversion “7"

    # INSTALLED means pfn refers to path in the container.
    # STAGEABLE means the executable can be staged into the container
    type "INSTALLED"

    #optional attribute to specify the container to use
    container "centos-pegasus"
  }
}

cont centos-pegasus{
     type "docker”

     image "docker:///centos:7"

     # optional site attribute to tell pegasus which site tar file
     # exists. useful for handling file URL's correctly
     image_site "optional site"

     # environment to be set when the job is run in the container
     # only env profiles are supported
     profile env "JAVA_HOME" "/opt/java/1.6"
}
...
```

container

Reference to the container to use. Multiple transformation can refer to same container

type

Can be either docker or singularity

image

URL to image in a docker|singularity hub
OR
to an existing docker image exported as a tar file or singularity image

Data Management for Containers

- Users can refer to container images as
  - Docker or Singularity Hub URL’s
  - Docker Image exported as a TAR file and available at a server , just like any other input dataset.
- If an image is specified to be residing in a hub
  - The image is pulled down as a tar file as part of data stage-in jobs in the workflow
  - The exported tar file is then shipped with the workflow and made available to the jobs
  - Motivation: Avoid hitting Docker/Singularity Hub repeatedly for large workflows
- Pegasus worker package is not required to be pre-installed in the container
  - If a matching worker package is not installed, the required worker package is installed at runtime when container starts

Container Execution Model

Containerized jobs are launched via Pegasus Lite

- Container image is put in the job directory along with input data.
- Loads the container if required on the node (applicable for Docker)
- Run a script in the container that sets up Pegasus in the container and launches user application
- Shut down the container (applicable for Docker)
- Ship out the output data generated by the application
- Cleanup the job directory

- Traditional shared-fs approach does not support containers.

Directories Mounted

- Only the job directory where PegasusLite places the inputs is mounted in the container
  - Docker – Mounted as /scratch
  - Singularity – Mounted as /srv

- PegasusLite ensures that user application is launched in the directory mounted
  - Consistent with the Pegasus model of ensuring that user job is launched in directory where it’s input data exists.

User Running in the Container

- Singularity containers always run in user space.

- Docker
  - Pegasus before launching the user application
    - Creates the user in the container that is the same as the user under which the job is launched by the Local Resource Manager on the remote node
  - Why do we do this?
    - By default, Docker runs user application as root
    - Not recommended for HPC environment
    - Creates problems with staging the outputs created in the container

Reference

- Documentation
  - https://pegasus.isi.edu/documentation/containers.php
- Example
  - https://github.com/pegasus-isi/montage-workflow-v2/
  - Script example-dss-containers.sh will run the montage workflow jobs in a container pulled from the singularity hub

Pegasus est. 2001

Automate, recover, and debug scientific computations.

Thank You

Questions?

Mats Rynge
rynge@isi.edu
USC Viterbi
School of Engineering
Information Sciences Institute

Meet our team

Ewa Deelman

Karan Vahi

Mats Rynge

Rajiv Mayani

Rafael Ferreira da Silva